# SIMATIC S5

# S5-115U
# Programmable Controller

**Manual**

**CPU 941-7UB11**

**CPU 942-7UB11**

**CPU 943-7UB11 and CPU 943-7UB21**

**CPU 944-7UB11 and CPU 944-7UB21**

***EWA 4NEB 811 6130-02b***

**Edition 04**

STEP®, SINEC® and SIMATIC® are registered trademarks of Siemens AG. LINESTRA® is a registered trademark of the OSRAM Company. IBM® is a registered trademark of the International Business Machines Corporation.
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.

**© Siemens AG 1991**

*EWA 4NEB 812 6130-02b*

Preface

Introduction

System Overview 1

Technical Description 2

Installation Guidelines 3

PLC System Start-Up and Program Test 4

Error Diagnostics 5

Addressing/Address Assignments 6

Introduction to STEP 5 7

STEP 5 Operations 8

Interrupt Processing 9

Analog Value Processing 10

Integral Blocks 11

Communications Capabilities 12

Integral Real-Time Clock 13

Reliability, Availability and Safety of Electronic Control Equipment 14

Technical Specifications 15

Appendices A/B/C/D/E

List of abbreviations

Index

# Contents


Page

**Preface** . . . . . xv

**Introduction** . . . . . xvii

**1 System Overview** . . . . . 1-1

1.1 Application . . . . . 1-1

1.2 System Components . . . . . 1-2
1.2.1 Power Supply . . . . . 1-2
1.2.2 Central Processing Units . . . . . 1-3
1.2.3 Input and Output Modules . . . . . 1-3
1.2.4 Intelligent Input/Output Modules . . . . . 1-4
1.2.5 Communications Processors . . . . . 1-4

1.3 Expansion Capability . . . . . 1-4
1.3.1 Centralized Configuration . . . . . 1-5
1.3.2 Distributed Configuration . . . . . 1-5

1.4 Communications Systems . . . . . 1-5

1.5 Operator-Process Communication, Monitoring, and Programming . . . . . 1-5

1.6 Software . . . . . 1-6

**2 Technical Description** . . . . . 2-1

2.1 Modular Design . . . . . 2-1

2.2 Functional Units . . . . . 2-3

2.3 Power Supply Modules . . . . . 2-6

2.4 Central Processing Units . . . . . 2-7

2.5 Operating Modes . . . . . 2-14
2.5.1 STOP Mode . . . . . 2-14
2.5.2 Restart Characteristics . . . . . 2-14
2.5.3 RUN Mode . . . . . 2-17
2.5.4 Restart Characteristics and Cyclic Operation . . . . . 2-17

2.6 Measuring and Estimating the Scan Time and Setting the Scan Monitoring Time . . . . . 2-21
2.6.1 Measuring the Scan Time . . . . . 2-21
2.6.2 Estimating the Scan Time . . . . . 2-22
2.6.3 Setting the Scan Monitoring Time . . . . . 2-27

Page

2.7 Accessories . . . . . 2.- 27
2.7.1 Backup Battery . . . . . 2.- 28
2.7.2 Memory Submodules . . . . . 2.- 28
2.7.3 Programmers (PG) . . . . . 2.- 29
2.7.4 Operator Panels (OP) . . . . . 2.- 29
2.7.5 Printers (PT) . . . . . 2.- 29

**3 Installation Guidelines** . . . . . 3.- 1

3.1 Mounting Rack . . . . . 3.- 1
3.1.1 Central Controller (CC) . . . . . 3.- 1
3.1.2 Expansion Unit (EU) . . . . . 3.- 8

3.2 Mechanical Installation . . . . . 3.- 13
3.2.1 Installing the Modules . . . . . 3.- 13
3.2.2 Installing Fans . . . . . 3.- 16
3.2.3 Dimension Drawings . . . . . 3.- 17
3.2.4 Cabinet Installation . . . . . 3.- 18
3.2.5 Centralized Configurations . . . . . 3.- 19
3.2.6 Distributed Configuration . . . . . 3.- 20
3.2.7 Other Possible Configurations . . . . . 3.- 28

3.3 Wiring . . . . . 3.- 29
3.3.1 Connecting the PS 951 Power Supply Module . . . . . 3.- 29
3.3.2 Connecting Digital Modules . . . . . 3.- 30
3.3.3 Front Connectors . . . . . 3.- 31
3.3.4 Simulator . . . . . 3.- 32
3.3.5 Connecting the Fan Subassembly . . . . . 3.- 33

3.4 General Configuration . . . . . 3.- 33
3.4.1 Power Supply . . . . . 3.- 33
3.4.2 Electrical Installation with Field Devices . . . . . 3.- 35
3.4.3 Connecting Nonfloating and Floating Modules . . . . . 3.- 40

3.5 Wiring Arrangement, Shielding and Measures against Electromagnetic Interference . . . . . 3.- 42
3.5.1 Running Cables Inside and Outside a Cabinet . . . . . 3.- 42
3.5.2 Running Cables Outside Buildings . . . . . 3.- 43
3.5.3 Equipotential Bonding . . . . . 3.- 44
3.5.4 Shielding Cables . . . . . 3.- 45
3.5.5 Special Measures for Interference-Free Operation . . . . . 3.- 46

Page

**4 PLC System Start-Up and Program Test** . . . . . 4 - 1

4.1 Prerequisites for Starting Up a PLC . . . . . 4 - 1

4.2 Steps for System Start-Up . . . . . 4 - 1
4.2.1 Overall Reset . . . . . 4 - 1
4.2.2 Transferring the Program . . . . . 4 - 3
4.2.3 Determining the Retentive Feature of Timers, Counters and Flags . . . . . 4 - 5

4.3 Testing the Program . . . . . 4 - 7
4.3.1 Starting the Program . . . . . 4 - 7
4.3.2 Search . . . . . 4 - 8
4.3.3 "Program Check" Test Function . . . . . 4 - 8
4.3.4 STATUS/STATUS VAR Test Function . . . . . 4 - 9
4.3.5 FORCE Outputs and Variables . . . . . 4 - 11

4.4 Special Features of the CPUs with Two Serial Interfaces . . . . . 4 - 12

4.5 Notes on the Use of Input/Output Modules . . . . . 4 - 13

4.6 System Start-Up . . . . . 4 - 14
4.6.1 Notes on Configuring and Installing a System . . . . . 4 - 14
4.6.2 System Start-Up Procedure . . . . . 4 - 15

**5 Error Diagnostics** . . . . . 5 - 1

5.1 Interrupt Analysis . . . . . 5 - 2
5.1.1 "ISTACK" Analysis . . . . . 5 - 2
5.1.2 Meaning of the ISTACK Displays . . . . . 5 - 6
5.1.3 LED Error Signalling . . . . . 5 - 9
5.1.4 Error Messages When Using Memory Submodules (only in the case of CPU 943/944) . . . . . 5 - 10

5.2 Program Errors . . . . . 5 - 11
5.2.1 Determining an Error Address . . . . . 5 - 12
5.2.2 Program Trace with the Block Stack ("BSTACK") Function (not possible on the PG 605U programmer) . . . . . 5 - 15

5.3 Other Causes of Malfunction . . . . . 5 - 16

5.4 System Parameters . . . . . 5 - 16

| | | Page |
|---|---|---|
| **6** | **Addressing/Address Assignments** | 6 - 1 |
| 6.1 | Address Structure | 6 - 1 |
| 6.1.1 | Digital Module Addresses | 6 - 1 |
| 6.1.2 | Analog Module Addresses | 6 - 1 |
| 6.2 | Slot Address Assignments | 6 - 1 |
| 6.2.1 | Fixed Slot Address Assignments | 6 - 2 |
| 6.2.2 | Variable Slot Address Assignments | 6 - 3 |
| 6.3 | Handling Process Signals | 6 - 7 |
| 6.3.1 | Accessing the PII | 6 - 8 |
| 6.3.2 | Accessing the PIQ | 6 - 9 |
| 6.3.3 | Direct Access | 6 - 10 |
| 6.4 | Address Allocation on the Central Processing Units | 6 - 11 |
| **7** | **Introduction to STEP 5** | 7 - 1 |
| 7.1 | Writing a Program | 7 - 1 |
| 7.1.1 | Methods of Representation | 7 - 1 |
| 7.1.2 | Operand Areas | 7 - 3 |
| 7.1.3 | Circuit Diagram Conversion | 7 - 3 |
| 7.2 | Program Structure | 7 - 4 |
| 7.2.1 | Linear Programming | 7 - 4 |
| 7.2.2 | Structured Programming | 7 - 5 |
| 7.3 | Block Types | 7 - 7 |
| 7.3.1 | Organization Blocks (OBs) | 7 - 8 |
| 7.3.2 | Program Blocks (PBs) | 7 - 11 |
| 7.3.3 | Sequence Blocks (SBs) | 7 - 11 |
| 7.3.4 | Function Blocks (FBs) | 7 - 11 |
| 7.3.5 | Data blocks (DBs) | 7 - 16 |
| 7.4 | Program Execution | 7 - 18 |
| 7.4.1 | RESTART Program Execution | 7 - 18 |
| 7.4.2 | Cyclic Program Execution | 7 - 20 |
| 7.4.3 | Time-Controlled Program Execution | 7 - 20 |
| 7.4.4 | Interrupt-Driven Programming Execution | 7 - 22 |
| 7.4.5 | Handling Programming Errors and PLC Malfunctions | 7 - 23 |
| 7.5 | Processing Blocks | 7 - 25 |
| 7.5.1 | Modifying the Program | 7 - 25 |
| 7.5.2 | Modifying Blocks | 7 - 25 |
| 7.5.3 | Compressing the Program Memory | 7 - 25 |
| 7.6 | Number Representation | 7 - 26 |

Page

**8 STEP 5 Operations** . . . . . 8 - 1

8.1 Basic Operations . . . . . 8 - 1
8.1.1 Boolean Logic Operations . . . . . 8 - 2
8.1.2 Set/Reset Operations . . . . . 8 - 7
8.1.3 Load and Transfer Operations . . . . . 8 - 10
8.1.4 Timer Operations . . . . . 8 - 15
8.1.5 Counter Operations . . . . . 8 - 25
8.1.6 Comparison Operations . . . . . 8 - 30
8.1.7 Arithmetic Operations . . . . . 8 - 31
8.1.8 Block Call Operations . . . . . 8 - 32
8.1.9 Other Operations . . . . . 8 - 38

8.2 Supplementary Operations . . . . . 8 - 39
8.2.1 Load Operation . . . . . 8 - 40
8.2.2 Enable Operation . . . . . 8 - 41
8.2.3 Bit Test Operations . . . . . 8 - 42
8.2.4 Digital Logic Operations . . . . . 8 - 44
8.2.5 Shift Operations . . . . . 8 - 48
8.2.6 Conversion Operations . . . . . 8 - 50
8.2.7 Decrement/Increment . . . . . 8 - 52
8.2.8 Disable/Enable Interrupt . . . . . 8 - 53
8.2.9 Processing Operation . . . . . 8 - 54
8.2.10 Jump Operations . . . . . 8 - 57
8.2.11 Substitution Operations . . . . . 8 - 59

8.3 System Operations . . . . . 8 - 65
8.3.1 Set Operations . . . . . 8 - 65
8.3.2 Load and Transfer Operations . . . . . 8 - 66
8.3.3 Jump Operation . . . . . 8 - 69
8.3.4 Arithmetic Operation . . . . . 8 - 70
8.3.5 Other Operations . . . . . 8 - 71

8.4 Condition Code Generation . . . . . 8 - 73

8.5 Sample Programs . . . . . 8 - 76
8.5.1 Momentary-Contact Relay (Edge Evaluation) . . . . . 8 - 76
8.5.2 Binary Scaler . . . . . 8 - 77
8.5.3 Clock (Clock-Pulse Generator) . . . . . 8 - 78
8.5.4 Delay Times . . . . . 8 - 79

Page


**9 Interrupt Processing . . . . . 9 - 1**

9.1 Programming Interrupt Blocks . . . . . 9 - 1

9.2 Calculating Interrupt Response Times . . . . . 9 - 3

9.3 Process Interrupt Generation with the 434-7 Digital Input Module . . . 9 - 5
9.3.1 Function Description . . . . . 9 - 5
9.3.2 Start-Up . . . . . 9 - 5
9.3.3 Initialization in Restart OBs . . . . . 9 - 5
9.3.4 Reading in the Process Signals . . . . . 9 - 7
9.3.5 Programming Example for Interrupt Processing . . . . . 9 - 8

9.4 Interrupt Processing with the Digital Input/Output Module 6ES5 485-7LA11 . . . . . 9 - 10
9.4.1 Function Description . . . . . 9 - 10
9.4.2 Operating the Module with Alarm Processing . . . . . 9 - 11
9.4.3 Operating the Module without Alarm Processing . . . . . 9 - 17
9.4.4 Notes on Characteristics of Inputs and Outputs . . . . . 9 - 18

**10 Analog Value Processing . . . . . 10 - 1**

10.1 Principle of Operation of Analog Input Modules . . . . . 10 - 1

10.2 Analog Input Module 460-7LA12 . . . . . 10 - 3
10.2.1 Connecting Transducers to the 460-7LA12 Analog Input Module . . . . 10 - 4
10.2.2 Putting Analog Module 460-7LA12 into Operation . . . . . 10 - 13

10.3 Analog Input Module 460-7LA13 . . . . . 10 - 16

10.4 Analog Input Module 465-7LA13 . . . . . 10 - 19
10.4.1 Connecting Transducers to the 465-7LA13 Analog Input Module . . . . 10 - 20
10.4.2 Starting Up the 465-7LA13 Analog Input Module . . . . . 10 - 24

10.5 466-3LA11 Analog Input Module . . . . . 10 - 28
10.5.1 Connecting Transducers to the 466-3LA11 Analog Input Module . . . . 10 - 29
10.5.2 Start-Up of the 466-3LA11 Analog Input Module . . . . . 10 - 33

10.6 Representation of the Digital Input Value . . . . . 10 - 39

10.7 Wirebreak Signal and Sampling for Analog Input Modules . . . . . 10 - 51

10.8 Principle of Operation of Analog Output Modules . . . . . 10 - 54
10.8.1 Connecting Loads to Analog Output Modules . . . . . 10 - 56
10.8.2 Digital Representation of an Analog Value . . . . . 10 - 58

10.9 Analog Value Matching Blocks FB250 and FB251 . . . . . 10 - 60

10.10 Example of Analog Value Processing . . . . . 10 - 64

Page


**11 Integral Blocks** . . . . . 11 - 1

11.1 Integral Function Blocks . . . . . 11 - 2
11.1.1 Conversion Blocks . . . . . 11 - 2
11.1.2 Arithmetic Blocks . . . . . 11 - 3
11.1.3 Data Handling Blocks . . . . . 11 - 5
11.1.4 The Integral "COMPR" Block . . . . . 11 - 28
11.1.5 Integral FB "DELETE" . . . . . 11 - 30

11.2 Organization Blocks . . . . . 11 - 32
11.2.1 OB31 Scan Time Triggering . . . . . 11 - 32
11.2.2 OB160 Variable Time Loop . . . . . 11 - 32
11.2.3 OB251 PID Control Algorithm . . . . . 11 - 33
11.2.4 OB254 Read In Digital Input Modules (CPU 944 Only) . . . . . 11 - 45
11.2.5 OB255 Transfer the Process Output Image (PIQ) to the Output Modules (CPU 944 Only) . . . . . 11 - 45

11.3 DB1: Initializing Internal Functions . . . . . 11 - 46
11.3.1 Configuration and Default Settings for DB1 . . . . . 11 - 46
11.3.2 Setting the Addresses for the Parameter Error Code in DB1 (An example of how to set the parameters correctly) . . . . . 11 - 47
11.3.3 How to Assign Parameters in DB1 . . . . . 11 - 48
11.3.4 Rules for Setting Parameters in DB1 . . . . . 11 - 49
11.3.5 How to Recognize and Correct Parameter Errors . . . . . 11 - 50
11.3.6 Transferring the DB1 Parameters to the PLC . . . . . 11 - 53
11.3.7 Reference Table for Initializing DB1 . . . . . 11 - 54
11.3.8 DB1 Programming Example . . . . . 11 - 56

**12 Communications Capabilities** . . . . . 12 - 1

12.1 Data Interchange . . . . . 12 - 1
12.1.1 Interprocessor Communication Flags . . . . . 12 - 1
12.1.2 Page Frame Addressing . . . . . 12 - 7

12.2 SINEC L1 Local Area Network . . . . . 12 - 7
12.2.1 Principle of Operation of the SINEC L1 Local Area Network . . . . . 12 - 8
12.2.2 Coordinating Data Interchange in the Control Program . . . . . 12 - 9
12.2.3 Assigning Parameters to the S5-115U for Data Interchange . . . . . 12 - 12

12.3 Point-to-Point Connection . . . . . 12 - 16
12.3.1 Connecting a Communications Partner . . . . . 12 - 17
12.3.2 Setting Parameters and Operation . . . . . 12 - 18

12.4 ASCII Driver (for CPU 943/944 with Two Serial Interfaces Only) . . . . . 12 - 20
12.4.1 Data Traffic . . . . . 12 - 21
12.4.2 Coordination Bytes . . . . . 12 - 23
12.4.3 Mode . . . . . 12 - 24
12.4.4 ASCII Parameter Set . . . . . 12 - 26
12.4.5 Assigning Parameters . . . . . 12 - 29
12.4.6 Sample Program for ASCII Driver . . . . . 12 - 30

Page


12.5 Communications Link Using the 3964/3964R Communications Protocol (for CPU 944 with Two Serial Ports Only) . . . . . 12 - 38
12.5.1 Data Interchange over the SI 2 Interface . . . . . 12 - 40
12.5.2 Assigning a Mode Number (System Data Word 55, $EA6E_H$) . . . . . 12 - 41
12.5.3 Assigning the Driver Number for a Communications Link . . . . . 12 - 42
12.5.4 Transmission . . . . . 12 - 42
12.5.5 Sample Program for Transmitting Data . . . . . 12 - 53

**13 Integral Real-Time Clock** . . . . . 13 - 1

13.1 Setting the System Data Parameters . . . . . 13 - 1
13.2 Structure of the Clock Data Area . . . . . 13 - 6
13.3 Structure of the Status Word . . . . . 13 - 10
13.4 Battery Backup of the Hardware Clock . . . . . 13 - 12
13.5 Programming the Integral Clock . . . . . 13 - 13

**14 Reliability, Availability and Safety of Electronic Control Equipment** . . . . . 14 - 1

14.1 Reliability . . . . . 14 - 1
14.1.1 Failure Characteristics of Electronic Devices . . . . . 14 - 2
14.1.2 Reliability of SIMATIC S5 Programmable Controllers and Components . . . . . 14 - 2
14.1.3 Failure Distribution . . . . . 14 - 3

14.2 Availability . . . . . 14 - 4

14.3 Safety . . . . . 14 - 5
14.3.1 Types of Failures . . . . . 14 - 5
14.3.2 Safety Measures . . . . . 14 - 6

14.4 Summary . . . . . 14 - 7

**15 Technical Specifications** . . . . . 15 - 1

15.1 General Technical Specifications . . . . . 15 - 3

15.2 Description of Modules . . . . . 15 - 5
15.2.1 Mounting Racks (CRs, ERs) . . . . . 15 - 5
15.2.2 Power Supply Modules . . . . . 15 - 9
15.2.3 Central Processing Units . . . . . 15 - 14
15.2.4 Digital Input Modules . . . . . 15 - 20
15.2.5 Digital Output Modules . . . . . 15 - 31
15.2.6 Digital Input/Output Module . . . . . 15 - 45
15.2.7 Analog Input Modules . . . . . 15 - 46
15.2.8 Analog Output Modules . . . . . 15 - 52
15.2.9 Intelligent Input/Output Modules . . . . . 15 - 58
15.2.10 Communications Processors . . . . . 15 - 59
15.2.11 Interface Modules . . . . . 15 - 60
15.2.12 The 313 Watchdog Module . . . . . 15 - 64

15.3 Accessories . . . . . 15 - 65

# Appendices


Page

**A Operations List** . . . . . A - 1

A.1 Explanation of the Operations List . . . . . A - 1
A.2 Basic Operations . . . . . 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 Maintenance** . . . . . B - 1

B.1 Changing Fuses . . . . . B - 1
B.2 Installing or Changing Battery . . . . . B - 1
B.2.1 Removing the Battery . . . . . B - 2
B.2.2 Installing the Battery . . . . . B - 2
B.2.3 Battery Disposal . . . . . B - 3
B.3 Changing the Fan Filter . . . . . B - 3
B.4 Replacing the Fan Motor . . . . . B - 4

**C Module Slots** . . . . . C - 1

C.1 Connector Pin Assignment for Power Supply Module . . . . . C - 1
C.2 Connector Pin Assignment of the CPUs . . . . . C - 2
C.3 Connector Pin Assignment for CPs and Intelligent I/Os . . . . . C - 3
C.4 Connector Pin Assignment for Digital and Analog Input/Output Modules . . . . . C - 4
C.5 Connector Pin Assignment for Interface Modules . . . . . C - 5
C.5.1 Connector Pin Assignment of the Symmetrical and Serial EU Interface Modules . . . . . C - 5
C.5.2 Connector Pin Assignment of the Symmetrical and Serial CC Interface Modules . . . . . C - 6
C.5.3 Connector Pin Assignment of the IM 305/IM 306 Interface Modules . . . . . C - 7
C.6 Connector Pin Assignment of the ER 701-3 Mounting Rack . . . . . C - 8
C.7 Legend for Connector Pin Assignment . . . . . C - 11

**D Active and Passive Faults in Automation Equipment / Guidelines for Handling Electrostatic Sensitive Devices** . . . . . D - 1

**E SIEMENS Addresses Worldwide** . . . . . E - 1

**List of Abbreviations**

**Index**

# Preface

The S5-115U is a programmable controller for the lower and mid performance ranges. It meets all the demands made of a modern programmable controller.
The performance capability of the S5-115U has recently been enhanced. In addition to increases in speed, the new generation of CPUs also offers uniform and user-friendly handling.

To put the controller to optimum use, you require a certain amount of detailed information. This manual presents all this information in an organized manner.

We have also been able to improve the quality of the manual with the help of your corrections and improvement suggestions. A proforma for further corrections and improvement suggestions is included at the end of the manual. You can help us to improve the next edition.

**Important Changes and Additions in this Manual:**

- Detailed and uniform description of the CPU operating modes and the STARTUP characteristics (Chapter 2)
- Taking account of EMC-oriented cable laying in the design of a controller (Chapter 3)
- Operator-oriented presentation of Chapter 4 ("PLC System Start-Up and Program Test")
- Improved representation of the structure of the STEP 5 programming language (Chapter 7)
- Additional "Interrupt Processing" section (Chapter 9)
- New version of the "Analog Value Processing" section (Chapter 10)
- Description of new integral organization blocks (Chapter 11)
- Simplified initialization of internal functions in DB1 (Section 11.3)

The idea behind all this has been to make sure you receive all the information you require for working with the S5-115U.

However, not all problems that might occur in the many and varied applications can be handled in detail in a manual. If you have a problem that is not discussed in the manual, contact your nearest SIEMENS office or representative. You will find a list in Appendix D.

# Introduction

The following pages contain information to help you familiarize yourself with the manual.

## Description of contents

The contents of the manual can be broken down subject-wise into a number of blocks:

- Description
  (System overview, technical description)
- Installation and operation
  (Installation guidelines, system start-up and program test, fault diagnostics, addressing)
- Programming instructions
  (Introduction to STEP 5, STEP 5 operations)
- Special capabilities
  (Analog value processing, integral blocks, communications)
- Technical specifications overview

You will find additional information in tabular form in the appendices.

Please use the forms at the back of the manual for any suggestions or corrections you may have and return the forms to us. This will help us to make the necessary improvements in the next edition.

## Training courses

Siemens offer comprehensive training facilities for users of SIMATIC S5.

Details can be obtained from your nearest Siemens office or representative.

## Reference literature

The manual contains a comprehensive description of the S5-115U. Subjects that are not specially related to the S5-115U have only been treated in brief, however. More detailed information is available in the following literature:

- **Programmable controls**
  Volume 1: Logic and sequence controls; from the control problem to the control program

  Günter Wellenreuther, Dieter Zastrow
  Brunswick 1987

  Contents:
  - Theory of operation of a programmable control system
  - Theory of logic control technology using the STEP 5 programming language for SIMATIC S5 programmable controllers.

  Order No.: ISBN 3-528-04464-0

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

  Hans Berger
  Siemens AG, Berlin and Munich 1989

  Contents:
  - STEP 5 programming language
  - Program scanning
  - Integral software blocks
  - I/O interfaces

  Order No.: ISBN 3-89578-022-7

Information on the programmable controller hardware is to be found in the following catalogues:

- ST 52.3 "S5-115U Programmable Controller"
- ST 57 "Standard Function Blocks and Driver Software for Programmable Controllers of the U Range"
- ST 59 "Programmers"
- ET 1.1 "ES 902 C Modular 19 in. Packaging System"
- MP 11 "Thermocouples; compensating boxes"

The relevant manuals are available for other components and modules (e.g. CPs and SINEC L1). Reference is made to these sources of information at various points in the manual.

The S5-115U programmable controller is designed to VDE 0160. The corresponding IEC and VDE (Association of German Electrical Engineers) standards are referred to in the text.

## Conventions

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

- The individual chapters can be quickly located by means of a thumb register.
- There is an overview containing the headings of the individual chapters at the beginning of the manual.
- Each chapter is preceeded by a breakdown to its subject matter.
  The individual chapters are subdivided into sections. Boldface type is used for further subdivisions.
- Figures and tables are numbered separately in each chapter. The page following the chapter breakdown contains a list of the figures and tables appearing in that particular chapter.

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

- A number of abbreviations have been used.
  Example: Programmer (PG)
- 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 or below the relevant table or figure.
- Lists are indicated by a black dot (•), as in this list for example, or with a dash (-).
  Instructions for operator actions are indicated by black triangles ( ).
- Cross references are shown as follows:
  "( Section 7.3.2)" refers to Section 7.3.2.
  No references are made to individual pages.
- All dimensions in drawings etc. are given in millimetres followed by inches in brackets. Example: 187 (7.29).
- Values may be represented as binary, decimal or hexadecimal numbers. The hexadecimal number system is indicated with a subscript (example $F000_H$)
- Information of special importance is enclosed in black-edged boxes:

**Warning**

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

Manuals can only describe the current version of the programmer. 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 "04". In the event of a revision, the edition number will be incremented by "1".
After revision of edition "03", the contents of edition "04" were updated.

## Safety-Related Guidelines for the User

This document provides the information required for the intended use of the particular product. The documentation is written for technically qualified personnel.

Qualified personnel as referred to in the safety guidelines in this document as well as on the product itself are defined as follows.

- System planning and design engineers who are familiar with the safety concepts of automation equipment.
- Operating personnel who have been trained to work with automation equipment and are conversant with the contents of the document in as far as it is connected with the actual operation of the plant.
- Commissioning and service personnel who are trained to repair such automation equipment and who are authorized to energize, de-energize, clear, ground, and tag circuits, equipment, and systems in accordance with established safety practice.

### Danger Notices

The notices and guidelines that follow are intended to ensure personal safety, as well as protect the products and connected equipment against damage.

The safety notices and warnings for protection against loss of life (the users or service personnel) or for protection against damage to property are highlighted in this document by the terms and pictograms defined here. The terms used in this document and marked on the equipment itself have the following significance.

**Danger**

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

**Warning**

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

**Caution**

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

**Note**

contains important information about the product, its operation or a part of the document to which special attention is drawn.

### Proper Usage

**Warning**

- The equipment/system or the system components may only be used for the applications described in the catalog or the technical description, and only in combination with the equipment, components, and devices of other manufacturers as far as this is recommended or permitted by Siemens.
- The product will function correctly and safely only if it is transported, stored, set up, and installed as intended, and operated and maintained with care.

## 1 System Overview


1.1 Application . . . . . 1 - 1

1.2 System Components . . . . . 1 - 2
1.2.1 Power Supply . . . . . 1 - 2
1.2.2 Central Processing Units . . . . . 1 - 3
1.2.3 Input and Output Modules . . . . . 1 - 3
1.2.4 Intelligent Input/Output Modules . . . . . 1 - 4
1.2.5 Communications Processors . . . . . 1 - 4

1.3 Expansion Capability . . . . . 1 - 4
1.3.1 Centralized Configuration . . . . . 1 - 5
1.3.2 Distributed Configuration . . . . . 1 - 5

1.4 Communications Systems . . . . . 1 - 5

1.5 Operator-Process Communication, Monitoring, and Programming . . . . . 1 - 5

1.6 Software . . . . . 1 - 6

**Figures**

1-1. S5-115U Components . . . . . 1. - 2

# 1 System Overview

The SIMATIC® S5-115U programmable controller is used worldwide in almost all fields in a wide range of applications. Each of its modular components handles a specific task. Therefore, you can expand the system according to your needs. Three types of communications systems pass information among multiple controllers. The S5-115U system provides operator panels, monitoring devices, and various programmers to suit your needs. The STEP 5 programming language and an extensive software catalog make programming easy.

## 1.1 Application

Many different industries use the S5-115U. Even though each automation task is different, the S5-115U adapts optimally to the most varied jobs, whether they involve simple open-loop control or complex closed-loop control.

Present areas of application include the following:

- Automobile Industry
  Automatic drill, assembly and test equipment, painting facilities, shock absorber test bays
- Plastics Industry
  Blow, injection, and thermal molding machines, synthetics production systems
- Heavy Industry
  Molding equipment, industrial furnaces, rolling mills, automatic pit shaft temperature control systems
- Chemical Industry
  Proportioning and mixing systems
- Food and Beverages Industry
  Brewery systems, centrifuges
- Machinery
  Packing, woodworking, and custom-made machines, machine controls, machine tools, drilling mills, fault alarm centers, welding technology
- Building Services
  Elevator technology, climate control, ventilation, lighting
- Transport Systems
  Transport and sorting equipment, high-bay warehouses, conveyor and crane systems
- Energy, Gas, Water, Air
  Pressure booster stations, standby power supply, pump control, water and air treatment, filtering and gas recovery systems

## 1.2 System Components

The S5-115U system is made up of various modular components, as pictured in Figure 1-1. These components include the following:

- power supply module (PS)
- central processing unit (CPU)
- input and output modules (I/Os)
- intelligent input/output modules (IPs, WFs)
- communications processors (CPs)

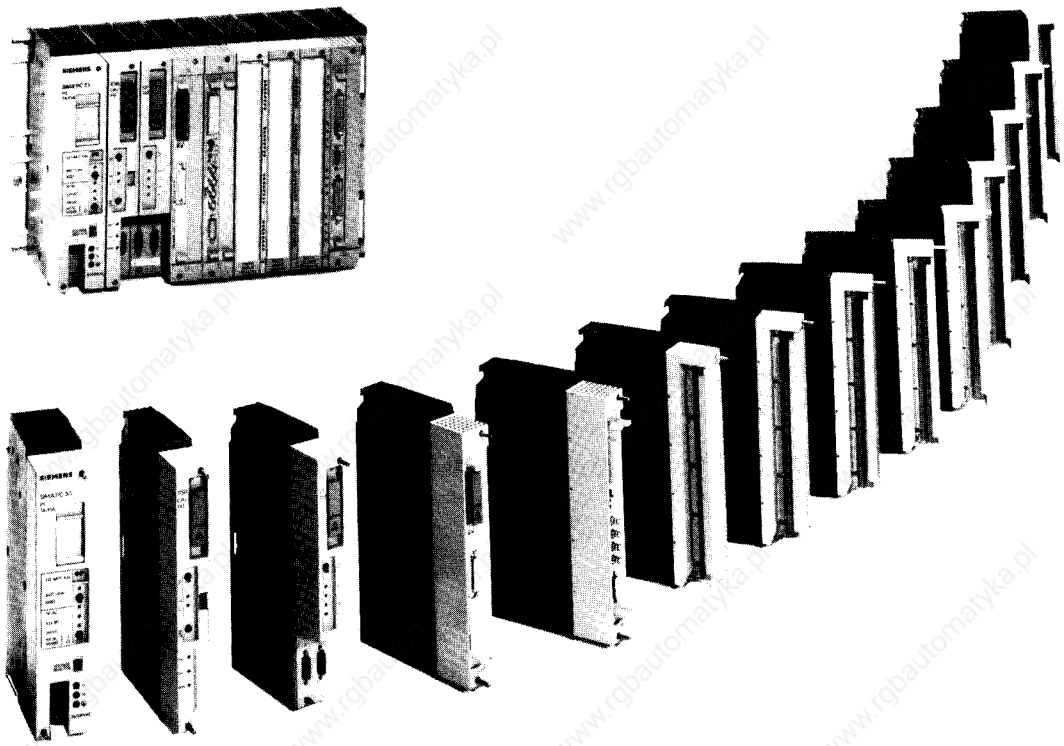


**Figure 1-1. S5-115U Components**

### 1.2.1 Power Supply

The power supply module (PS) converts the external power supply to the internal operating voltage. Supply voltages for the S5-115U include 24 V DC, 120 V AC, and 230 V AC.

Screw-type terminals connect the power supply lines to the bottom of the PS. Three maximum output currents are available. Choose 3 A, 7 A, or 15 A according to the number of modules you have or according to their power consumption. A fan is not necessary for output currents up to 7 A.

A lithium battery backs up the program memory and the internal retentive flags, timers and counters in the event of a power failure. An LED signals battery failure. If you change the battery when the power is shut off, connect a back-up voltage from an outside source to the sockets provided for this purpose on the power supply module.

### 1.2.2 Central Processing Units

The central processing unit (CPU) is the "brain" of the programmable controller. It executes the control program. Choose from the following four CPUs, depending on the degree of performance your S5-115U must have:

CPU 941, CPU 942, CPU 943 and CPU 944.

The more powerful the CPU you choose, the shorter are your program execution times and the larger the user memory. You can use CPUs 941 to 944 also for PID control - in conjunction with analog modules and PID control software - since the operating systems in these CPUs contain integral PID control algorithms. Sampling times from 100 ms are possible for a PID control loop. You can implement up to eight PID control loops.
CPU 943 and CPU 944 (each with two serial interfaces) offer further possibilities for controlling the process thanks to their integral hardware clock.

### 1.2.3 Input and Output Modules

Input and output modules are the interfaces to the sensors and actuators of a machine or controlled system.
The following features make S5-115U modules easy to handle:
- fast installation
- mechanical coding
- large labeling areas

**Digital Modules**

Digital modules conform to the voltage and current levels of your machine. You do not have to adapt the existing level to the programmable controller. The S5-115U adapts itself to your machine.
Digital modules have the following convenient features:
- connection of signal lines via front connectors
- a choice of screw-type or crimp snap-in connections

**Analog Modules**

As a programmable controller's degree of performance increases, so does the significance of its analog value processing. The significance of the analog input and output modules increases accordingly.
Analog modules handle mainly closed-loop control tasks, such as automatic level, temperature, or speed control.

The S5-115U offers floating and non-floating analog input modules. They use one range card for every four channels to adapt the desired signal level.

This feature allows you to do the following:
- have up to four different measuring ranges on one module, depending on the number of channels a module has
- change the measuring ranges simply by exchanging range cards

Three analog output modules cover the various voltage or current ranges of analog actuators.

### 1.2.4 Intelligent Input/Output Modules

Counting rapid pulse trains, detecting and processing position increments, measuring time and speed, closed-loop control, and positioning are just a few of many time-critical jobs. The central processor of a programmable controller usually cannot execute such jobs fast enough in addition to its actual control task. The S5-115U provides intelligent input/output modules (IPs) to handle these time-critical jobs. Use these modules to handle measuring, closed-loop control, and open-loop control tasks rapidly in parallel to the program.
Most of the modules have their own processor to handle tasks independently. All these modules have a high processing speed and are easy to handle. Standard software puts them into operation.

### 1.2.5 Communications Processors

The S5-115U offers a number of special communications processors (CPs) to make communication easier between man and machine or machine and machine.

The two main groups of CPs are as follows:
- CPs for local area networks
- CPs for linking, signalling, and logging

## 1.3 Expansion Capability

If the connection capability of one central controller (CC) is no longer sufficient for your machine or system, increase the capacity with expansion units (EUs).

Interface modules connect a CC to EUs and connect EUs to each other. Choose an interface module suitable to the controller configuration you need.

### 1.3.1 Centralized Configuration

A centralized configuration allows you to connect up to three EUs to one CC. The interface modules for this purpose connect bus lines and supply voltage to the EUs. The EUs in such configurations therefore need no power supplies of their own. The cables between the individual controllers have a total maximum length of 2.5 m (8.2 ft.).

### 1.3.2 Distributed Configuration

A distributed configuration allows you to relocate expansion units nearer to the sensors and actuators of your machine.
Distributed configurations reduce cabling costs for these devices.

## 1.4 Communications Systems

Controller flexibility is critical to manufacturing productivity. Complex control tasks can be divided and distributed over several controllers to achieve the greatest flexibility possible.

Distribution offers the following advantages:
- small units that are easier to manage. You can plan, start up, diagnose, modify, and operate your system more easily, and observe the entire process more easily
- enhanced system availability because, if one unit fails, the rest of the system continues to function

Information must flow between distributed controllers to ensure the following:
- data exchange between programmable controllers
- central monitoring, operation, and control of manufacturing systems
- collection of management information such as production and warehousing data

For this reason, we offer the following communications facilities for the S5-115U programmable controller:
- Point-to-point connection with the CP 524 and CP 525 communications processors
- Local area network communications via the SINEC L1 network
- Industrial Ethernet
- PROFIBUS
- Point-to-point connection with the CPUs 943 and 944
- ASCII interface (in CPU 943 and CPU 944) for connecting printer, keyboard, etc.
- Computer connection with 3964/3964R protocol (in CPU 944)

## 1.5 Operator-Process Communication, Monitoring, and Programming

Today, users expect good process visualization with the capability to intervene where necessary. Previously, they had to hard wire indicating lights, switches, potentiometers, and pushbuttons, even for simple requirements. For more complex processes, they had to use expensive video display terminals. Inflexible or expensive solutions are a thing of the past.

In price and performance, the S5-115U offers you a graduated spectrum of operator panels and monitoring devices - from a small hand-held operator panel to a convenient color video display terminal.

The S5-115U enables you to react optimally to the most varied automation requirements, even where programming is concerned.

To help you with this, the following graduated and compatible spectrum of programmers is available:
- the economical PG 605U hand-held programmer
- the PG 635 in briefcase design with swing-up liquid crystal display
- PG 685 with CRT-based user friendliness
- PG 710
- PG 730
- PG 750
- PG 770

All the programmers feature high performance, simple handling, user-friendly operator prompting, and the standard, easily learned STEP 5 programming language.

## 1.6 Software

Until now, prices for hardware components tended to drop constantly and prices for software tended to increase. The reasons were as follows:
- the processes to be automated became more and more complex
- safety requirements increased
- personnel costs increased
- ergonomic demands increased

Siemens has put an end to this trend. SIMATIC provides the following three solutions to keep software costs down:
- the user-friendly STEP 5 programming language with its four methods of representation and convenient structuring capabilities
- an extensive software catalog
- user-friendly programmers

## 2 Technical Description


2.1 Modular Design . . . . . 2 .- 1

2.2 Functional Units . . . . . 2 .- 3

2.3 Power Supply Modules . . . . . 2. - 6

2.4 Central Processing Units . . . . . 2. - 7

2.5 Operating Modes . . . . . 2 .- 14
2.5.1 STOP Mode . . . . . 2. - 14
2.5.2 Restart Characteristics . . . . . 2. - 14
2.5.3 RUN Mode . . . . . 2. - 17
2.5.4 Restart Characteristics and Cyclic Operation . . . . . 2 - 17

2.6 Measuring and Estimating the Scan Time and Setting the Scan Monitoring Time . . . . . 2. - 21
2.6.1 Measuring the Scan Time . . . . . 2. - 21
2.6.2 Estimating the Scan Time . . . . . 2. - 22
2.6.3 Setting the Scan Monitoring Time . . . . . 2 - 27

2.7 Accessories . . . . . 2. - 27
2.7.1 Backup Battery . . . . . 2 .- 28
2.7.2 Memory Submodules . . . . . 2. - 28
2.7.3 Programmers (PG) . . . . . 2 .- 29
2.7.4 Operator Panels (OP) . . . . . 2. - 29
2.7.5 Printers (PT) . . . . . 2. - 29

## Figures


2-1. The S5-115U (Central Unit) . . . . . 2 - 1
2-2. Schematic of the S5-115U . . . . . 2 - 3
2-3. Power Supply Module Control Panel . . . . . 2 - 6
2-4. Schematic Representation of CPU 941 and CPU 942 . . . . . 2 - 9
2-5. Schematic Representation of CPU 943 . . . . . 2 - 10
2-6. Schematic Representation of CPU 944 . . . . . 2 - 11
2-7. Front View of the Central Processing Units . . . . . 2 - 12
2-8. Control Panel of the Different CPUs . . . . . 2 - 13
2-9. Restart Characteristics of the CPU . . . . . 2 - 18
2-10. Cold Restart Characteristics After Power Restore . . . . . 2 - 19
2-11. Conditions for Changing the Operating Mode . . . . . 2 - 20
2-12. Subdivision of the Scan Time . . . . . 2 - 22
2-13. User Time ($T_A$) . . . . . 2 - 22
2-14. System Time . . . . . 2 - 25
2-15. Response Time . . . . . 2 - 26


## Tables


2-1. CPU Comparison . . . . . 2 - 7
2-2. Execution Times in µsec. (Rounded off) . . . . . 2 - 8
2-3. Operating Mode LEDs . . . . . 2 - 13
2-4. System Data Area; List of All Addressable I/O Words
(DI=Digital Input Byte, DQ=Digital Output Byte,
AI=Analog Input Byte, AQ=Analog Output Byte) . . . . . 2 - 15
2-5. Subdivision of the User Time . . . . . 2 - 23
2-6. Ready Delay Times of the Various I/O Modules . . . . . 2 - 24
2-7. System Time . . . . . 2 - 25
2-8. Available Memory Submodules . . . . . 2 - 28

# 2 Technical Description

This chapter describes the design and principle of operation of an S5-115U with accessories.

## 2.1 Modular Design

The S5-115U consists of various functional units that can be combined to suit the particular problem.

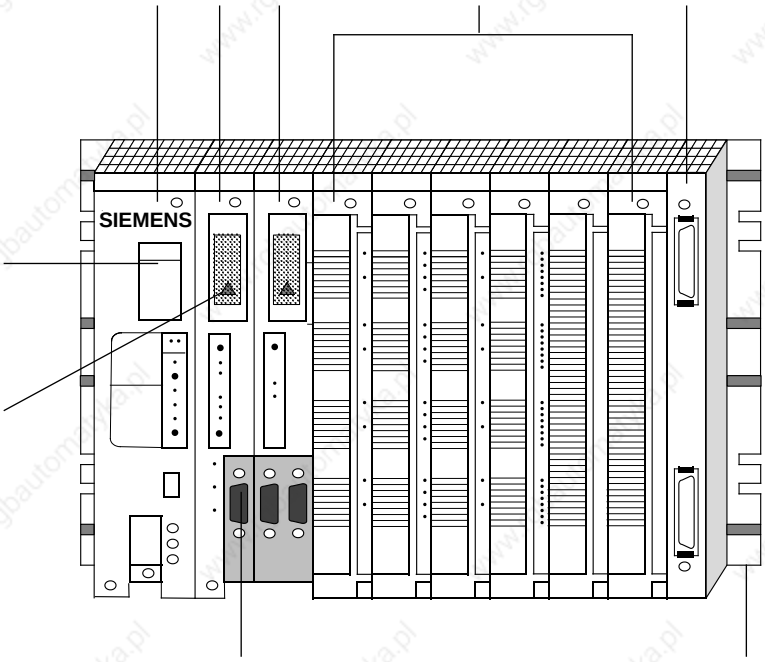



**Figure 2-1. The S5-115U (Central Unit)**

The numbered information below briefly describes the most important components of the S5-115U.

**Power Supply Module (PS 951)**
The PS 951 power supply module generates the operating voltage for the PLC from the 120 V AC/230 V AC or 24 V DC power system voltages. This module uses a battery or an external power supply to back up the RAM.
The PS 951 power supply module also performs monitoring and signalling functions.

**Central Processing Unit (CPU)**
The central processing unit reads in input signal states, processes the control program, and controls outputs. In addition to program scanning functions, the CPU provides internal flags, timers and counters. You can preset the restart procedure and diagnose errors using the CPU's LEDs. Use the Overall Reset switch on the CPU to delete the RAM contents.
Use a programmer or a memory submodule to transfer the control program to the CPU.

**Communications Processors (CP)**
Communications processors can be used in the S5-115U for communication between man and machine and between machines. Communications processors perform the following functions:
- operator monitoring and control of machine functions or process sequences
- reporting and listing of machine and process states

You can connect various peripheral devices to these processors, e.g. printers, keyboards, CRTs and monitors as well as other controllers and computers.

**Input/Output Modules (I/Os)**
- Digital input modules adapt digital signals, e.g. from pressure switches or BERO® proximity switches, to the internal signal level of the S5-115U.
- Digital output modules convert the internal signal level of the S5-115U into digital process signals, e.g. for relays or solenoid valves.
- Analog input modules adapt analog process signals, e.g. from transducers or resistance thermometers, to the S5-115U, which functions digitally.
- Analog output modules convert internal digital values of the S5-115U to analog process signals, e.g. for speed controllers.

**Interface Modules (IM)**
The S5-115U is installed on mounting racks with a specific number of mounting locations (slots). A configuration comprising power supply, CPU, and input/output modules is called a central controller. If the slots on the central controller's mounting rack are insufficient, you can install expansion units (systems without CPUs) on additional mounting racks. Interface modules connect an expansion unit to a central controller.

**Mounting Racks**
A mounting rack consists of an aluminium rail to which all the modules are fastened mechanically. It has one or two backplanes that connect the modules to each other electrically.

**Serial Interface**
You can connect the following at this interface:
- Programmer
- Operator panel
- SINEC L1 bus terminal

**Memory submodule**

**Battery compartment**

Not represented:

**Operating System Submodule (only CPU 944)**
As well as the PLC operating system, this submodule also contains driver blocks for the second interface. They are loaded into the user memory of the interface after power restore.

**Intelligent Input/Output Modules (IPs)**
Intelligent input/output modules are available for handling the special tasks:

- counting rapid pulse trains
- measuring and processing positioning increments
- measuring speed and time
- controlling temperatures and drives, and so on.
  Intelligent input/output modules generally have their own processor and thus off-load the CPU. Consequently, they can process measuring and open- and closed-loop control tasks quickly while the CPU handles other jobs.

## 2.2 Functional Units

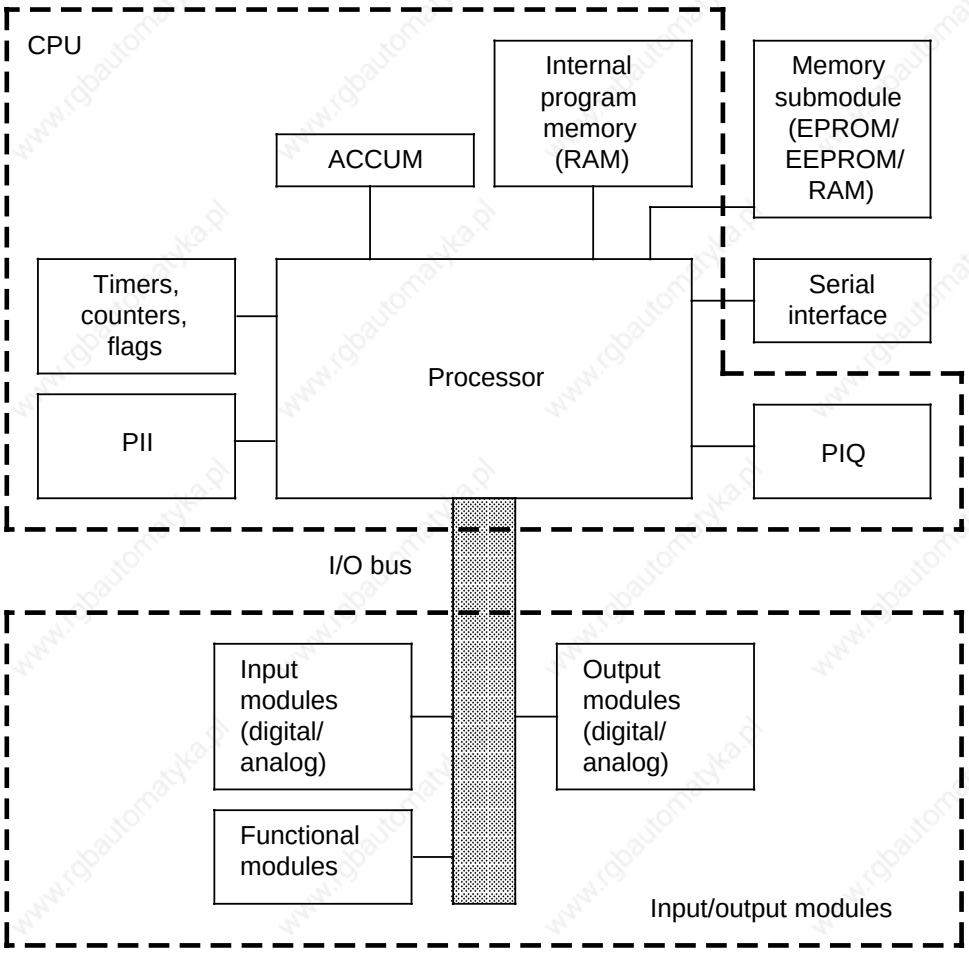



**Figure 2-2. Schematic of the S5-115U**

**Program Memory (Internal Program Memory, Memory Submodule)**

The control program is stored in the memory submodule or in the internal program memory (RAM). The CPU 943 and CPU 944 can hold the entire program in internal RAM.
To safeguard against losing the program, dump it in an external EPROM or EEPROM memory submodule. In contrast to these memory submodules, the internal RAM or a RAM memory submodule has the following characteristics:

- The memory contents can be changed quickly.
- User data can be stored and changed.
- When the power fails and there is no battery, the memory contents are lost.

**Process Images (PII, PIQ)**

Signal states of input and output modules are stored in the CPU in "process images". Process images are reserved areas in CPU RAM.
Input and output modules have separate images as follows:

- Process input image (PII)
  and
- Process output image (PIQ)

**Serial Interface**

You can connect programmer, operator panels and monitors at the serial interface. You can also connect the SINEC L1 local area network at the serial interface on all CPUs. You can order CPU 943 and CPU 944 with a second serial interface. The following additional functions are possible at this interface:

- Point-to-point connection to other programmable controllers
- ASCII driver for connecting printer, keyboard, etc.
- Integral real-time clock (see Chapter 13);

CPU 944 only:

- Communications link (3964/3964R line procedure; see Chapter 12)

**Timers, Counters and Flags**

Each CPU provides the control program with internal timers, counters and flags. Flags are memory locations for storing signal states. Timers, counters and flags can each be set as "retentive" (by area), i.e. their contents are not lost at POWER OFF. Memory areas whose contents are reset at POWER OFF are "non-retentive".

**Accumulator (ACCUM)**

The accumulator is an arithmetic register for loading, for example, internal times and counts. Comparison, arithmetic and conversion operations are also executed in the accumulator.

**Processor**

The processor calls statements in the program memory in sequence and executes them in accordance with the control program. It processes the information from the PII and takes into consideration the values of internal timers and counters as well as the signal states of internal flags.

**I/O Bus**

The I/O bus establishes the electrical connection for all signals that are exchanged between the CPU and the other modules in a central controller or an expansion unit.

**Memory Submodules**

The following three memory submodule types are available for the S5-115U to store the control program or to transfer the program to the PLC:

- **EPROM Submodules**
  Use an ultraviolet erasing device to delete the submodule's contents.
- **EEPROM Submodules**
  Program and erase EEPROM submodules on a programmer.
- **RAM Submodules**
  are used in addition to program storage to test a control program during system start-up. They should be used as program memories only when backup is guaranteed.

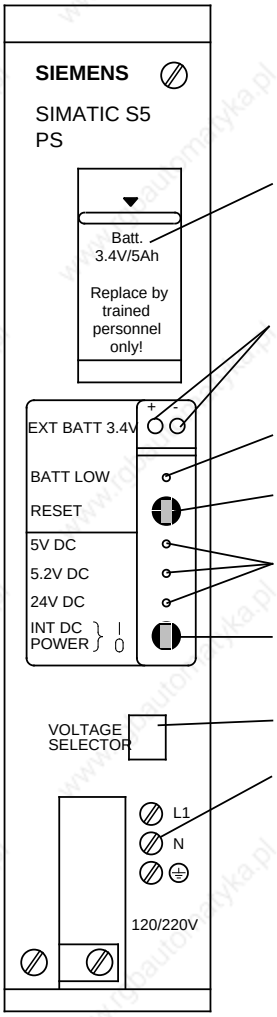
The individual submodules are available with different memory capacities. See the end of this section for a table of memory submodules you can use (see Accessories).

## 2.3 Power Supply Modules

Power supply modules generate the operational voltages for the PLC from the 120/230 V AC or 24 V DC mains supply and they provide backup for the RAM with a battery or an external power supply.
Power supply modules also execute monitoring and signalling functions.

You can set the following switches on the PS 951 power supply module:

- The Voltage Selector switch sets the line voltage at either 120 V AC or 230 V AC for AC modules. The PS 951 can also be operated with a 24 V DC power supply.
- The ON/OFF switch turns the operating voltages on or off.
- The RESET switch acknowledges a battery failure indication.




Battery compartment

Sockets for external 3, 4 to 9 V DC for backup (when battery is changed and power supply is shut off)

Battery failure indicator
The LED lights up under the following conditions:
- There is no battery.
- The battery has been installed incorrectly.
- The battery voltage has dropped below 2.8 V.

If the LED lights up, the "BAU" signal is sent to the CPU.

RESET switch
Use this switch to acknowledge a battery failure signal after you have installed a new battery. If you are operating the PS 951 power supply module without a battery, activate this switch to suppress the "BAU" signal.

Operating voltage displays
- +5 V supply voltage for the input/output modules
- +5.2 V supply voltage for PG 605U, OPs, BT 777 bus terminal
- +24 V for serial interface (20 mA current loop interface).

ON/OFF switch (I=ON, 0=OFF)
When the switch is in the "OFF" position, the operating voltages are disabled without interrupting the connected line voltage.

120 V AC/230 V AC Voltage Selector switch with transparent cover.

Screw-type terminals for connecting the line voltage

**Figure 2-3. Power Supply Module Control Panel**

## 2.4 Central Processing Units

Four CPU types are available for the S5-115U. Tables 2-6 and 2-7 show the most important CPU features.

**Table 2-1. CPU Comparison**

| | CPU 941 | CPU 942 | CPU 943 | CPU 944 |
|---|---|---|---|---|
| Execution time per<br>- 1000 statements<br>(see Appendix A for specific information) | Approx.<br>10 msec. | Approx.<br>10 msec. | Approx.<br>5 msec. | Approx.<br>1.5 msec. |
| Internal program memory (RAM) | 2 Kbytes | 10 Kbytes | 48 Kbytes | 96 Kbytes |
| Total program memory, maximum | 18 Kbytes* | 42 Kbytes* | 48 Kbytes | 96 Kbytes |
| Cycle monitoring time | Default approx. 500 ms, programmable | | | |
| Program scanning | Cyclic, interrupt-driven, time-controlled | | | |
| Address range, maximum (digital inputs) | 1024<br>I 0.0 to I 127.7 | | | |
| Address range, maximum (digital outputs) | 1024<br>Q 0.0 to Q 127.7 | | | |
| Address range, maximum (analog inputs) | 64<br>PW 128 to PW 254 | | | |
| Address range, maximum (analog outputs) | 64<br>PW 128 to PW 254 | | | |
| Flags | 1024, optionally<br>• all retentive<br>• half retentive<br>• all non-retentive | | | |
| Timers | 128, optionally<br>• all retentive<br>• half retentive<br>• all non-retentive | | | |
| Counters | 128, optionally<br>• all retentive<br>• half retentive<br>• all non-retentive | | | |
| Time range | 0.01 to 9990 s | | | |
| Counting range | 0 to 999 | | | |
| Operation set | Approx. 170 operations | | | |

* Sum from the internal program memory and submodule

**Table 2-1. CPU Comparison (Continued)**

| | CPU 941 | CPU 942 | CPU 943 | CPU 944 |
|---|---|---|---|---|
| S5-115U Closed-loop control | Yes | Yes | Yes | Yes |
| ASCII driver | No | No | Yes* | Yes* |
| Point-to-point connection (as master) | No | No | Yes* | Yes* |
| SINEC L1 | Yes | Yes | Yes | Yes |
| Communications link | No | No | No | Yes* |
| Realtime clock | No | No | Yes** | Yes** |

* Only at interface SI 2 in the case of CPUs with two serial interfaces

** Only in the case of CPUs with two serial interfaces

**Table 2-2. Execution Times in µsec. (Rounded off)**

| Operations | Execution Time in µsec. | | | |
|---|---|---|---|---|
| | CPU 941 | CPU 942 | CPU 943 | CPU 944 |
| Boolean logic operations | 1.6 | | 0.8 | |
| Load/Transfer operations (I, Q, F, T, C) | 1.6 | | 0.8 | |
| Comparison/Arithmetic operations | 1.6 | | 0.8 | |
| Jump/Conversion operations | 1.6 | | 0.8 | |
| Timer/Counter operations | 3.7 | | 1.8 | |
| Block Call operations | 1.6 to 6.7 | | 0.8 to 3.6 | |
| Load/Transfer operations (DW) | 2.2 to 3.9 | | 1.1 to 1.9 | |
| Substitution operations (formal operands)* | 160 | | | 3.6 |
| Load/Transfer operations (Periph., LIR, TIR, TNB)** | 70 to 126 | | | 4 |
| DO operations (DO DW, DO FW) | 134 to 162 | | | 3.6/2.6 |
| Bit Test operations | 159 | | | |

* plus execution time of the substituted operation

** plus transfer time (see Appendix A.2 and A.4)

### CPU 941 and CPU 942

The CPU 941 and the CPU 942 both contain a microprocessor and an application-specific integrated circuit (ASIC). The microprocessor handles all programmer interface module functions, processes interrupts and substitution operations and controls the S5 bus. The microprocessor also controls the ASIC that handles high-speed processing of STEP 5 operations. Besides the operating system memory, the CPU 941 and CPU 942 also contain an internal RAM that can be used to store the control program
(CPU 941: 2 Kbytes
CPU 942: 10 Kbytes).

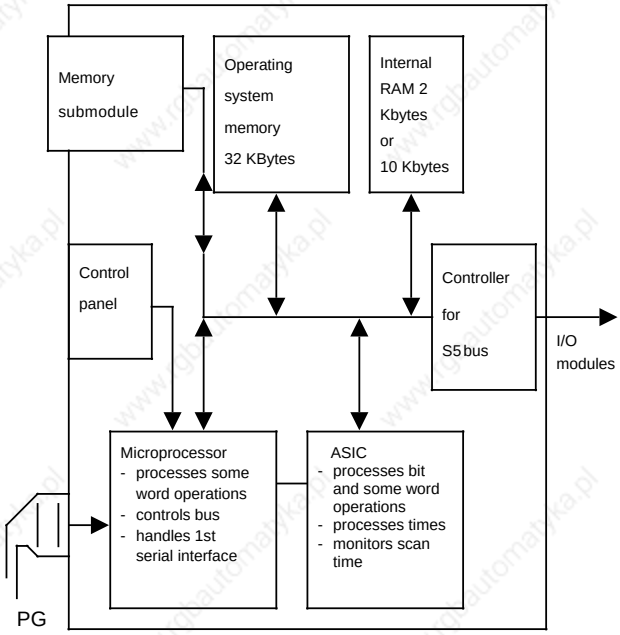



**Figure 2-4. Schematic Representation of CPU 941 and CPU 942**

**CPU 943**

The CPU 943 contains an application-specific integrated circuit (ASIC) and a microprocessor. The microprocessor handles all programmer interface module functions, processes interrupts and substitution operations and controls the S5 bus. The microprocessor also controls the ASIC that handles high-speed processing of STEP 5 operations. Besides the operating system memory, the CPU 943 also has an internal RAM (48 Kbytes) that can be used to store the control program.

The contents of memory submodules are copied to the internal RAM after POWER UP and after overall reset.

The CPU 943 can also be ordered with two interfaces. The second interface is controlled by a further microprocessor with its own operating system. This operating sytem is also stored in the operating system memory for the CPU. Programmers, operator panels (OPs) and SINEC L1 can be connected to the second interface; point-to-point connection via the SINEC L1 protocol is also possible. The CPU 943 operating system also supports the following additional functions:

- ASCII driver for data interchange over the second interface and for connecting peripheral devices (e.g. printers)
- Integral real-time clock.

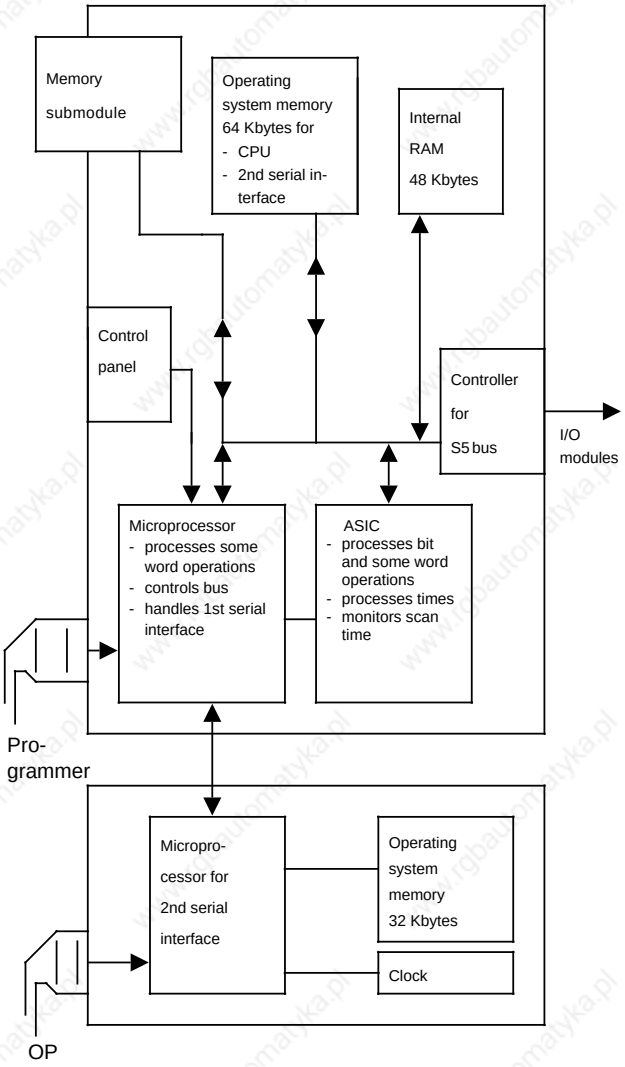



**Figure 2-5. Schematic Representation of CPU 943**

**CPU 944**

The CPU 944 contains two application-specific integrated circuits (ASICs) and a microprocessor.

The microprocessor handles all programmer interface module functions and processes interrupts. The microprocessor also controls the ASIC that handles high-speed processing of STEP 5 operations, monitors the scan time and controls I/O accesses. The CPU 944 also has an internal RAM (48 Kbytes) that can be used to store the control program.

The contents of memory submodules is copied to the internal RAM after POWER UP and after overall reset. The operating system is stored on a special memory submodule and can be replaced without opening the module.

The CPU 944 can also be ordered with two interfaces. The second interface is controlled by a further microprocessor with its own operating system. This operating system is also stored on the special memory submodule that can be replaced without opening the module. Programmers, operator panels (OPs) and SINEC L1 can be connected to the second interface; point-to-point connection via the SINEC L1 protocol is also possible. The CPU 944 operating system also supports the following additional functions:

- ASCII driver for data interchange over the second interface and for connecting peripheral devices (e.g. printers)
- Computer interface (3964(R) procedure) with operating system submodule for this purpose
- Integral real-time clock.

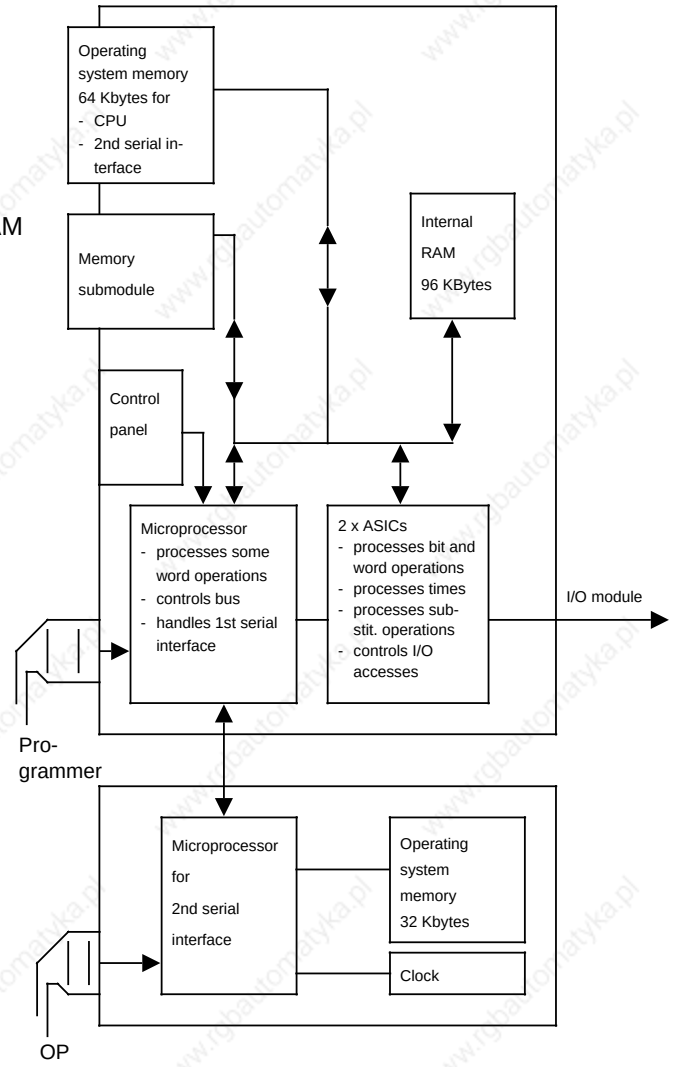


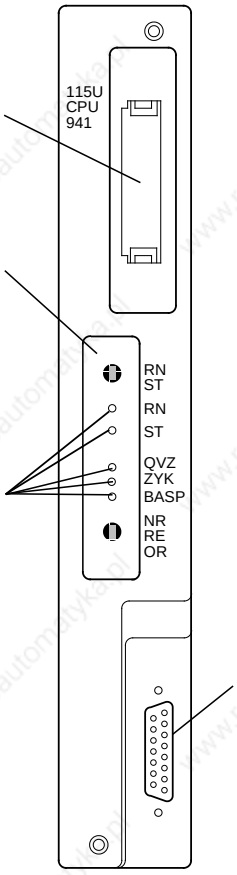

**Figure 2-6. Schematic Representation of CPU 944**

**Front Panels of the Central Processing Units**

The following operator functions are possible on the front panel of the CPUs:

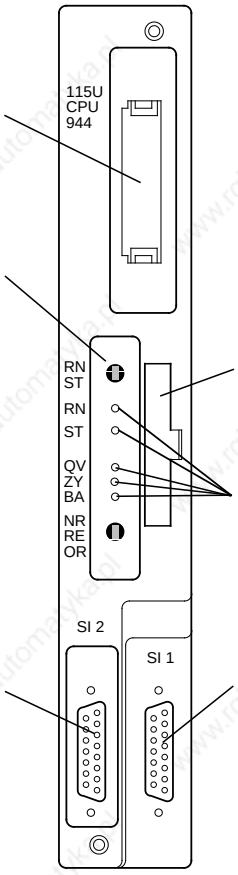
- Plug in a memory submodule
- Connect a programmer (PG) or an operator panel (OP)
- Connect SINEC L1
- Connect PLCs or devices of other manufacture
  - CPU 943/944: connection with ASCII driver or point-to-point connection (master function)
  - only in the case of the CPU 944: computer interface (3964(R) procedure)
- Set the operating mode
- Preset retentive feature
- Perform Overall Reset
- Change the operating system submodule (only CPU 944)

LEDs indicate the current CPU status. Figure 2-7 compares the individual CPUs.




View of CPU 941/942

Receptacle for memory submodule
Control panel
Connection sockets for PG, OP or SINEC L1 LAN

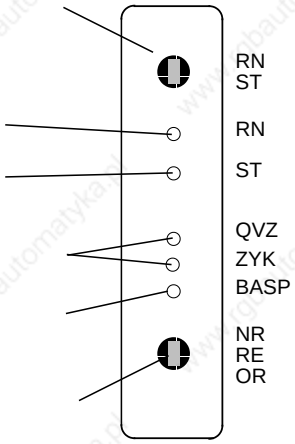



View of CPU 943/944

Connection sockets for PG, OP or SINEC L1 LAN, point-to-point connection (master function), ASCII driver, (CPU 944 only:) computer interface
Operating mode and error LEDs
Receptacle for operating system submodule (only CPU 944)

**Figure 2-7. Front View of the Central Processing Units**

The CPU controls are arranged in a panel. Figure 2-8 shows the control panel of the different CPUs.




Mode selector STOP/RUN
RUN LED
STOP LED

Switch for the following RESTART settings:
- nonretentive presetting (NR)
  retentive presetting (RE)
- overall reset (OR)

Error LEDs (QVZ, ZYK)
BASP (output disable); outputs of the output modules are not enabled

**Figure 2-8. Control Panel of the Different CPUs**

### Meaning of the LEDs

Two LEDs on the control panel of the CPU indicate the status of the CPU ( and in Figure 2-8). Table 2-3 lists the possible indications.
A flashing or flickering red LED indicates PLC malfunctions (see Chapter 5).

**Table 2-3. Operating Mode LEDs**

| Red LED | Green LED | Meaning |
|---|---|---|
| ✳ | ✳ | CPU is in cold restart routine or in RESTART mode |
| ✳ | ○ | STOP mode |
| ○ | ✳ | RUN mode |
| ○ | ○ | Program check running |

## 2.5 Operating Modes

Use the mode selector to set the "STOP" (ST) or "RUN" (RN) mode. The CPU executes the "RESTART" mode automatically between "STOP" and "RUN".

### 2.5.1 STOP Mode

The program is not scanned in STOP mode. The values of the timers, counters, flags and process images that were current when the CPU went into the STOP state are maintained. Output modules are disabled (signal state "0"). The BASP (output disable) LED lights up. The BASP signal is cancelled only after OB21 or OB22 (RESTART) have been processed.

### 2.5.2 Restart Characteristics

Everything that takes place between
- a STOP RUN transition (manual cold restart) or
- a POWER UP RUN transition (automatic cold restart after power up)

is referred to as restart characteristics.

Two phases can be distinguished during restart:
- The ***cold restart routine*** (PLC cannot be directly influenced)
- The actual ***RESTART*** (PLC characteristics can be controlled in RESTART OBs (OB21 and OB22)).

**Cold Restart Routine**

The following applies while the CPU runs the cold restart routine:
- The BASP LED lights up;
  - The status of the error LEDs remains unchanged during manual cold restart
  - The error LEDs light up momentarily during automatic cold restart after power up
- Outputs display signal "0" if all output modules are disabled
- All inputs and outputs in the process I/O image display signal "0"
- Scan time monitoring is inactive.

During the cold restart routine, the processor ascertains the configuration of the I/O modules and stores this information. This procedure is described in detail in the following.

To establish the configuration of the I/O modules, the processor checks the full address area of the input/output modules word by word. If it addresses a module over an I/O word (=2 bytes), the processor "notes" this word by setting the bit allocated to it in a special memory area called the system data area. This bit is only set by the processor if both I/O bytes of an I/O word are addressable.
The processor uses a system data word (SD) to check 16 I/O words (=32 I/O bytes).

Using this method, the processor determines the bytes of the process I/O image to be updated during process I/O image transfer. Table 2-4 lists all relevant system data words in the system data area.
If, for instance, I/O bytes 24 ***and*** 25 (=I/O word 24)

- ***can be read***, bit 4 is set in ***system data word (SD) 16***;
- ***can be written to***, bit 4 is set in ***system data word 20***.

**Table 2-4. System Data Area; List of All Addressable I/O Words (DI=Digital Input Byte, DQ=Digital Output Byte, AI=Analog Input Byte, AQ=Analog Output Byte)**

| SD | Bit | | | | | | | | | | | | | | | | Abs. Addr. of the Syst. Data Word |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | |
| | 7 | | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 | | |
| 16 | DI | 14 | DI | 12 | DI | 10 | DI | 8 | DI | 6 | DI | 4 | DI | 2 | DI | 0 | $EA20_H$ |
| | DI | 15 | DI | 13 | DI | 11 | DI | 9 | DI | 7 | DI | 5 | DI | 3 | DI | 1 | |
| | DI | 30 | DI | 28 | DI | 26 | DI | 24 | DI | 22 | DI | 20 | DI | 18 | DI | 16 | $EA21_H$ |
| | DI | 31 | DI | 29 | DI | 27 | DI | 25 | DI | 23 | DI | 21 | DI | 19 | DI | 17 | |
| ⋮ | | | | | | | | | | | | | | | | | ⋮ |
| 20 | DQ | 14 | DQ | 12 | DQ | 10 | DQ | 8 | DQ | 6 | DQ | 4 | DQ | 2 | DQ | 0 | $EA28_H$ |
| | DQ | 15 | DQ | 13 | DQ | 11 | DQ | 9 | DQ | 7 | DQ | 5 | DQ | 3 | DQ | 1 | |
| | DQ | 30 | DQ | 28 | DQ | 26 | DQ | 24 | DQ | 22 | DQ | 20 | DQ | 18 | DQ | 16 | $EA29_H$ |
| | DQ | 31 | DQ | 29 | DQ | 27 | DQ | 25 | DQ | 23 | DQ | 21 | DQ | 19 | DQ | 17 | |
| ⋮ | | | | | | | | | | | | | | | | | ⋮ |
| 24 | AI | 14 | AI | 12 | AI | 10 | AI | 8 | AI | 6 | AI | 4 | AI | 2 | AI | 0 | $EA30_H$ |
| | AI | 15 | AI | 13 | AI | 11 | AI | 9 | AI | 7 | AI | 5 | AI | 3 | AI | 1 | |
| | AI | 30 | AI | 28 | AI | 26 | AI | 24 | AI | 22 | AI | 20 | AI | 18 | AI | 16 | $EA31_H$ |
| | AI | 31 | AI | 29 | AI | 27 | AI | 25 | AI | 23 | AI | 21 | AI | 19 | AI | 17 | |
| ⋮ | | | | | | | | | | | | | | | | | ⋮ |
| 28 | AQ | 14 | AQ | 12 | AQ | 10 | AQ | 8 | AQ | 6 | AQ | 4 | AQ | 2 | AQ | 0 | $EA38_H$ |
| | AQ | 15 | AQ | 13 | AQ | 11 | AQ | 9 | AQ | 7 | AQ | 5 | AQ | 3 | AQ | 1 | |
| | AQ | 30 | AQ | 28 | AQ | 26 | AQ | 24 | AQ | 22 | AQ | 20 | AQ | 18 | AQ | 16 | $EA39_H$ |
| | AQ | 31 | AQ | 29 | AQ | 27 | AQ | 25 | AQ | 23 | AQ | 21 | AQ | 19 | AQ | 17 | |
| ⋮ | | | | | | | | | | | | | | | | | ⋮ |
| 31 | AQ | 110 | AQ | 108 | AQ | 106 | AQ | 104 | AQ | 102 | AQ | 100 | AQ | 98 | AQ | 96 | $EA3E_H$ |
| | AQ | 111 | AQ | 109 | AQ | 107 | AQ | 105 | AQ | 103 | AQ | 101 | AQ | 99 | AQ | 97 | |
| | AQ | 126 | AQ | 124 | AQ | 122 | AQ | 120 | AQ | 118 | AQ | 116 | AQ | 114 | AQ | 112 | $EA3F_H$ |
| | AQ | 127 | AQ | 125 | AQ | 123 | AQ | 121 | AQ | 119 | AQ | 117 | AQ | 115 | AQ | 113 | |

**Programmable Restart Delay at Cold Restart and After Power Restore**

If you want to delay checking the module configuration because, for example, switching the voltage to a remotely connected EU is delayed, you must modify system data word 126 ($EAFC_H$) in one of the following ways:

- With the DISPL ADDR programmer function (only permissible when the CPU is in the STOP mode!)
- With STEP 5 operations in the control program (only in FBs).

In any event, the restart delay will only become effective after the next POWER OFF POWER ON transition and remains effective until the next modification to this system data word. After Overall Reset, the default applies ($0000_H$, i.e. no delay). One unit in system data word 126 corresponds to a restart delay of 1 ms; the longest possible delay is 65535 ms ($FFFF_H$).

**Example:** Programming a restart delay of approximately one minute

| STL FB98 | Explanation |
|---|---|
| :L KH EA60<br>:T BS 126<br>:BE | Restart delay of 60,000 ms (1 minute) |

> **Note**
>
> If no backup battery has been inserted in the power supply module (or if the inserted battery is defective) and the control program is stored on an E(E)PROM submodule, the restart will be delayed by approximately one second.

**Restart**

While the CPU is in RESTART, the following applies:

- The fault LEDs are dark; the RUN, STOP and BASP LEDs light up
- All output modules are disabled (outputs show signal "0")
- The PII is not yet updated; evaluation of the inputs is only possible with direct I/O access (L PY../L PW..)

  Example:
  ```
  L PW 0
  T IW 0
  A I  0.0
     :
  ```
- Scan time monitoring is inactive
- The relevant RESTART OB is processed (in the case of manual cold restart of OB21, in the case of automatic cold restart of OB22 - if the mode selector is at "RN")
- Timers are processed
- Interrupt OBs (OB2 to OB6) and timed-interrupt OBs (OB10 to OB13) are only processed if the interrupts are explicitly enabled (RA operation).

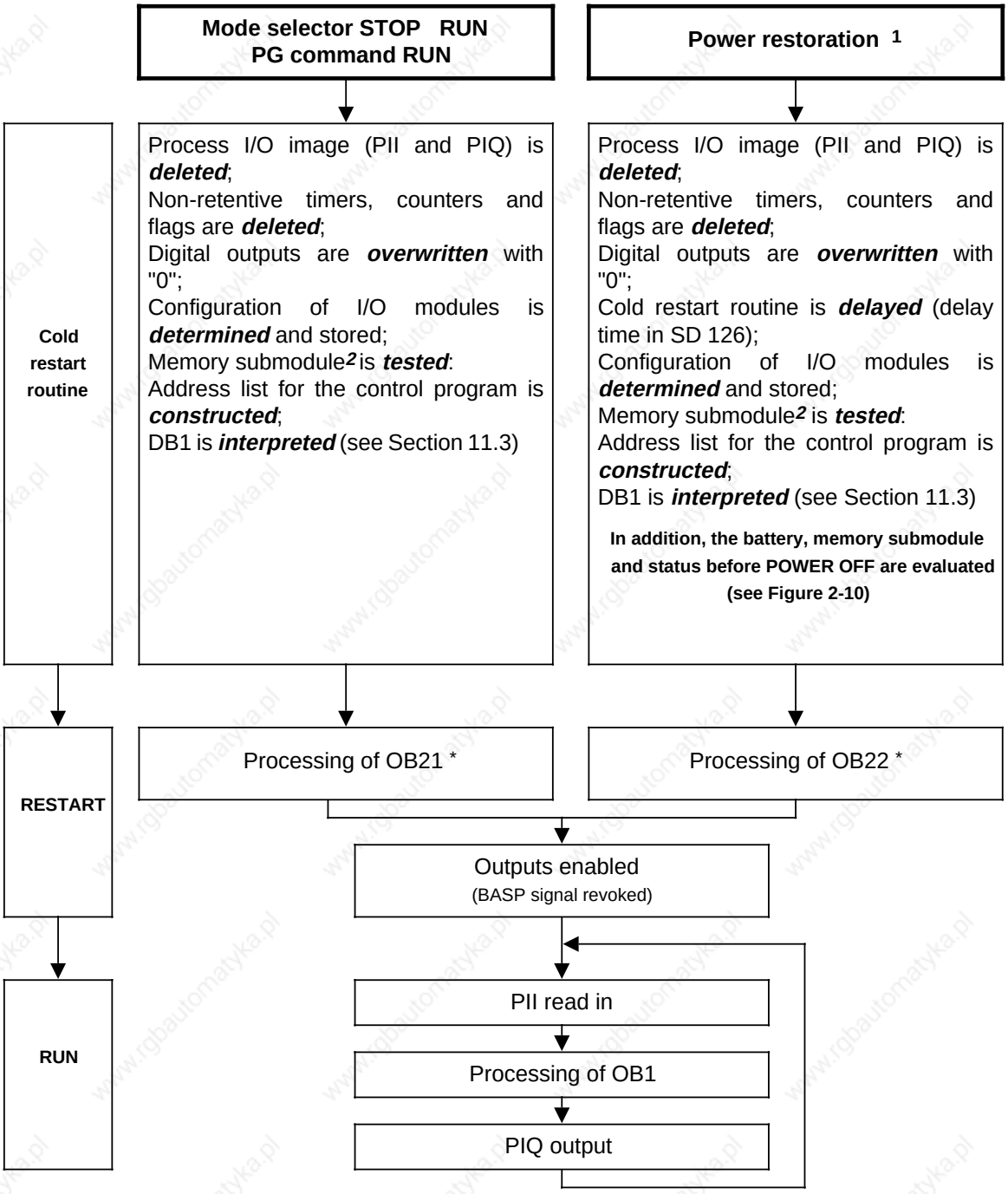
### 2.5.3 RUN Mode

After the CPU operating system has run the RESTART program, it starts cyclic program scanning (OB1).
The input signals at the input modules are scanned cyclically and mapped to the PII; the interprocessor communication input flags (see Section 12.1.1) are updated. The control program processes this information together with the current flag, timer and counter data. The control program comprises a succession of individual statements. The processor fetches these statements one by one from program memory and executes them. The results are written to the process output image (PIQ).

It is possible to react quickly to signal changes even during cyclic program scanning by:
- Programming organization blocks to service interrupts
- Using operations with direct I/O access (e.g. LPW, TPW)
- Multiple programming of direct I/O scans in the control program.

### 2.5.4 Restart Characteristics and Cyclic Operation

The following figures give an overview of the restart characteristics of the CPUs and of cyclic operation. They also show how the restart characteristics depend on the state of the backup battery and they indicate the conditions for changing the operating mode.




1 If the PLC was in RUN at POWER OFF.

2 Please note the following differences:
- CPU 941/942: Program runs from memory submodule
- CPU 943/944: Program is loaded from the memory submodule into internal program memory and processed there

***** If OB21 or OB22 contains the RA (enable interrupt) operation, a central process interrupt is possible from this point. If this operation has not been used in the RESTART OB, interrupt and timed-interrupt OBs can only become effective after the RESTART OB has been processed.

**Figure 2-9. Restart Characteristics of the CPU**

### Cold Restart Characteristics After Power Restoration

Battery status, memory submodule and status before POWER OFF are evaluated as follows on cold restart after power restoration:

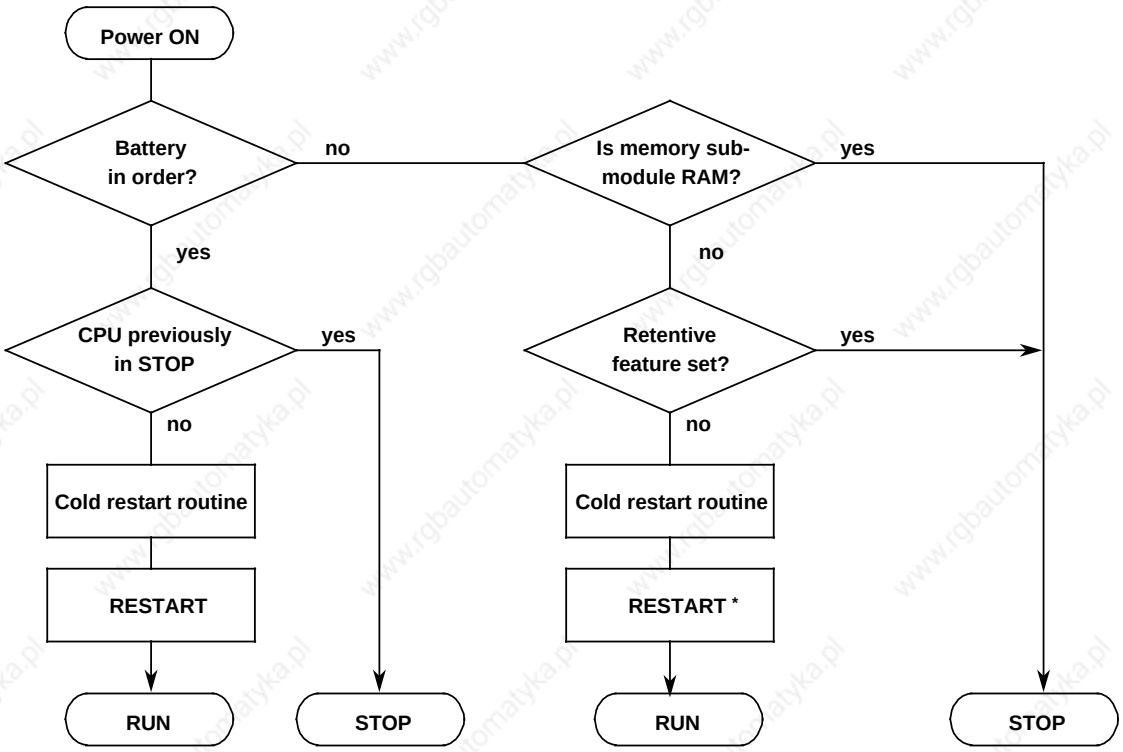



**Figure 2-10. Cold Restart Characteristics After Power Restoration**

## Changing the Operating Mode

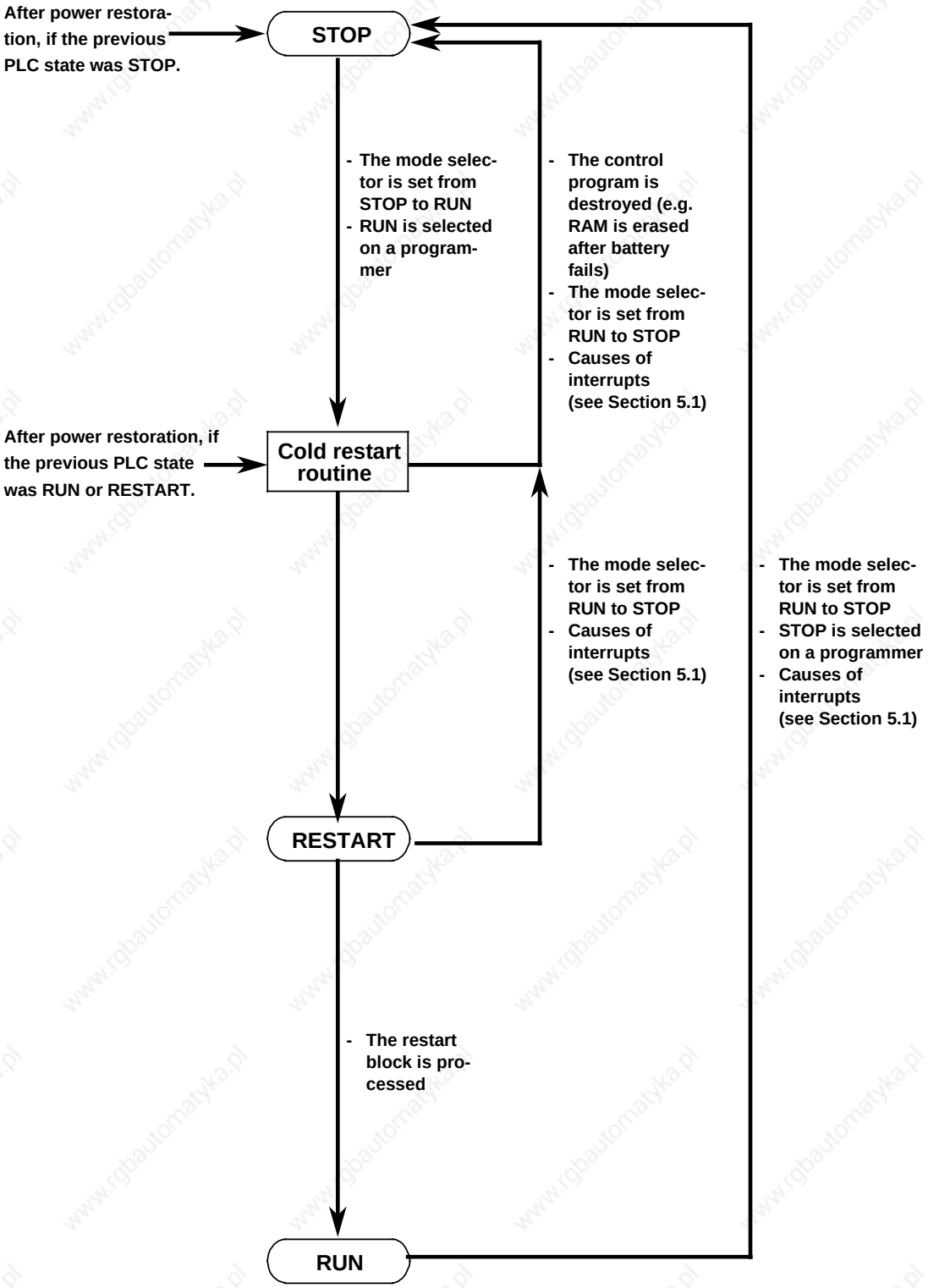



**Figure 2-11. Conditions for Changing the Operating Mode**

## 2.6 Measuring and Estimating the Scan Time and Setting the Scan Monitoring Time

### 2.6.1 Measuring the Scan Time

The scan time is measured by the CPU and stored in the system data area. You can access the current, the minimum and the maximum scan time in the control program at any time. The resolution of the scan time is one millisecond, and the range of scan time values extends from 0 to 32,767 (=7FFF$_H$) milliseconds. At the end of a scan cycle, after it has updated the process output image (PIQ) and the interprocessor communication flags, the operating system stores the scan time, i.e.:

- Current scan time in SD 121
- Maximum scan time in SD 122
- Minimum scan time in SD 123

If the scan time exceeds 32,767 milliseconds, bit 15 (which is the overflow bit) of the current scan time is set and entered in system data word SD 123 (maximum scan time). Scan time measurement begins anew in the next scan cycle.

> **Note**
>
> The contents of the watchdog timer are also entered in system data words 121 to 123 when the PLC stops and outputs the "ZYK" (scan time exceeded) message.

**Example**: Function block for measuring the scan time

| FB99 STL | Description |
|---|---|
| NAME :ZYKLUS-Z | |
| DECL :MINI I/Q/D/B/T/C: A BI/BY/W/D: W | |
| DECL :AKTU I/Q/D/B/T/C: A BI/BY/W/D: W | |
| DECL :MAXI I/Q/D/B/T/C: A BI/BY/W/D: W | |
| DECL :LOES I/Q/D/B/T/C: E BI/BY/W/D: BI | |
| : L RS 121 | |
| : T =AKTU | |
| : L RS 122 | |
| : T =MAXI | |
| : L RS 123 | |
| : T =MINI | |
| : AN =LOES | The LOES operand is used to reset system |
| : BEC | data words 121, 122 and 123 |
| : L KF +0 | (if LOES = 1). |
| : T RS 121 | |
| : T RS 122 | |
| : T RS 123 | |
| : BE | |

## 2.6.2 Estimating the Scan Time

The scan time has been divided into various units in the figures below to help you estimate the program runtime and thus the amount of time needed to scan the program.
The values shown below are only guidelines, and may differ depending on the configuration of the system involved.

The scan time is divided into user time and system time.

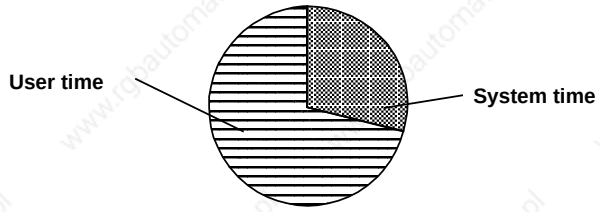



**Figure 2-12. Subdivision of the Scan Time**

Figure 2-13 shows how the user time is subdivided. Table 2-5 provides information on estimated times.

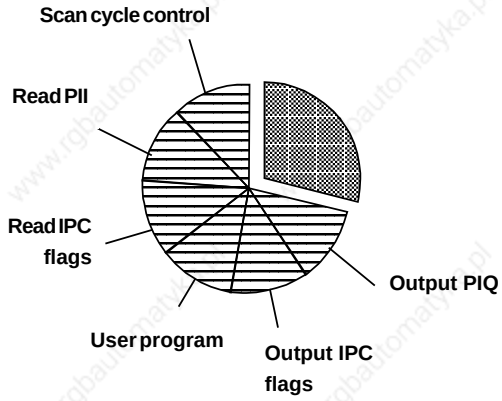



**Figure 2-13. User Time ($T_A$)**

**Table 2-5. Subdivision of the User Time**

| User Time $T_A$ | | |
|---|---|---|
| **Time required for** | **CPU** | **Time in µsec.** |
| Scan cycle control | CPU 941 | 160 |
| | CPU 942 | 160 |
| | CPU 943 | 160 |
| | CPU 944 | 210 |
| Reading the PII<br>n=No. of input bytes | CPU 941 | 140+<br>n · (30+module's Ready delay time*) |
| | CPU 942 | |
| | CPU 943 | |
| | CPU 944 | 60+<br>n · (1.7+module's Ready delay time*) |
| Reading the IPC flags<br>n=No. of IPC input flag bytes | CPU 941 | 530+ n · (44+module's Ready delay time*) |
| | CPU 942 | |
| | CPU 943 | |
| | CPU 944 | 60+<br>n · (1.7+module's Ready delay time*) |
| User programm (including OB2 to 5 and OB10 to 13) | | Sum of the execution times of all STEP 5 statements processed |
| Updating the PIQ<br>n=No. of output bytes | CPU 941 | 140+<br>n · (30+module's Ready delay time*) |
| | CPU 942 | |
| | CPU 943 | |
| | CPU 944 | 60+<br>n · (1.7+module's Ready delay time*) |
| Output IPC flags<br>n=No. of IPC output flag bytes | CPU 941 | 530+<br>n · (44+module's Ready delay time*) |
| | CPU 942 | |
| | CPU 943 | |
| | CPU 944 | 60+<br>n · (1.7+module's Ready delay time*) |

* See also Table 2-6

The Ready delay time is the time that elapses between the arrival of the Request signal in the module and the module's Ready signal.
The delay time depends on

- The Ready delay time of the module itself
- The interface module used and
- The length of the cable

In a distributed configuration, the communications link delay must also be taken into account. The signal propagation rate is 6 µsec./km, i.e. a distance-velocity lag of 2 x 6 µsec.=12 µsec. must be taken into account for a cable with a length of 1000 m (3,300 ft.).

If the CPU does not receive the Ready signal within 160 µsec., it stops and outputs the "QVZ" (time-out) error message.

**Table 2-6. Ready Delay Times of the Various I/O Modules**

| I/O Modules | Ready Delay Time in µsec. |
|---|---|
| Digital modules | 2 |
| Analog modules | 16 |
| 313 watchdog modules | 1 |
| IP 240 | 1 |
| IP 241 | 1 |
| IP 242 (Release A00) | 140 |
| IP 242 (Release A01) | 50 |
| IP 243 (Analog module) | 35 |
| IP 244 | 150 |
| IP 245 | 0.5 |
| IP 246 | 1.5 |
| IP 247 | 1.5 |
| IP 252 | 10 |
| WF 625 | 3 |
| CP 513-3M | 5 |
| CP 524 | 1 |
| CP 525 | 3 |
| CP 526 | 3 |
| CP 527 | 3 |
| CP 530 | 3 to 130 |
| CP 535 | 3 |
| CP 551 | 3 |
| CP 552 | 3 |
| CP 5430 | 1 |
| CP 143 | 3 |

Figure 2-14 shows the subdivision of the system time. The time values are listed in Table 2-7.

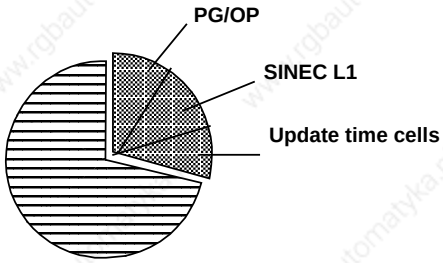



**Figure 2-14. System Time**

**Table 2-7. System Time**

| System Time | | |
|---|---|---|
| Time load caused by PG/OP | Approx. 6% of the user time ($T_A$) | |
| Time load caused by SINEC L1 | At SI 1: Up to 100% of the user time<br>At SI 2: Negligible | |
| Updating of the operating system timers<br><br>n=No. of timers active in the scan cycle | **CPU** | **Time in µsec.** |
| | CPU 941 | ($T_A$/10msec.) · (260+n · 0.8) |
| | CPU 942 | ($T_A$/10msec.) · (260+n · 0.8) |
| | CPU 943 | ($T_A$/10msec.) · (215+n · 0.4) |
| | CPU 944 | ($T_A$/10msec.) · (225+n · 0.4) |

**Response Time**

Response time is the period between the input signal change and the output signal change.

This time is typically the sum of the following elements (see also Figure 2-15):
- the inherent delay of the input module
- the program scan time

The delay of the output modules is negligible.

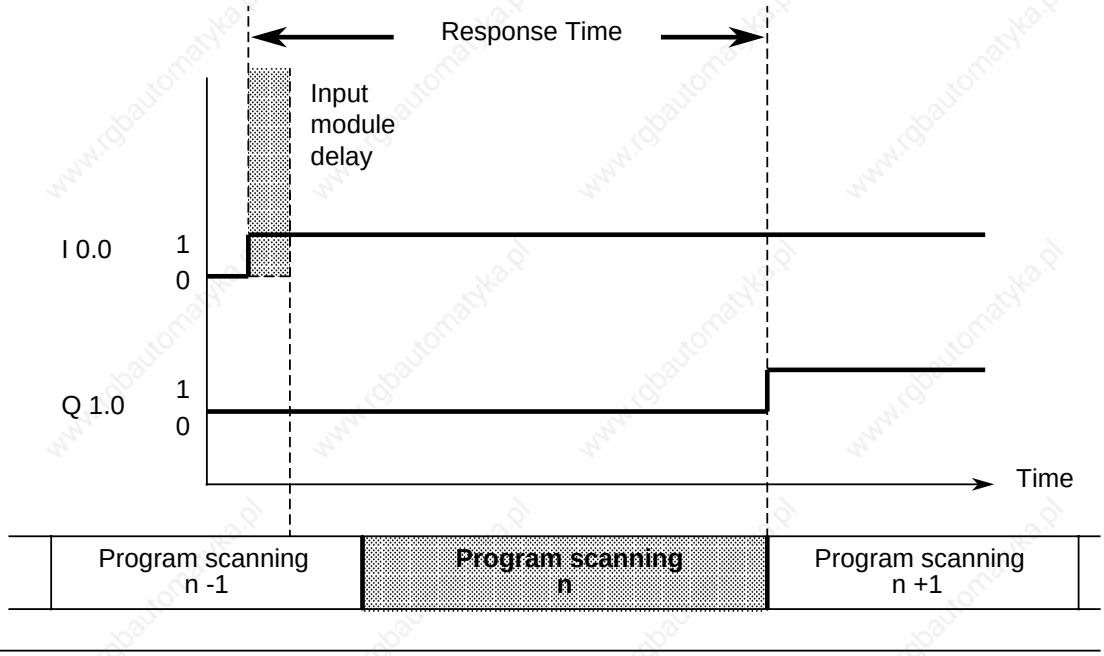



**Figure 2-15. Response Time**

Under worst-case circumstances, the response time is twice the program scan time.

### 2.6.3 Setting the Scan Monitoring Time

The scan time comprises the duration of the cyclic program. At the beginning of each program scan, the processor starts a monitoring time (cycle trigger). This monitoring time is preset to approximately 500 ms. If the scan trigger is not initiated again within this time - for example, as a result of programming an endless loop in the control program or as a result of a fault in the CPU - the PLC goes to STOP and disables all output modules. If the control program is very complex, and the monitoring time can be exceeded, you should restart the monitoring time in the control program (OB13, see Chapter 11).

There are two possible methods for changing the default scan monitoring time:
- Initialization in DB1 (see Section 11.3)
- STEP 5 operations.

You can set the scan monitoring time up to 2.55 s (KF=+255) without having to restart the monitoring time.

If you want to change the default scan monitoring time (approximately 500 msec.) via STEP 5 operations, you must transfer a factor for this purpose to system data word 96. The CPU operating system interprets this factor as a multiple of 10 msec.

**Example:**
The scan monitoring time is to be set to 100 msec. after every manual cold restart and after automatic cold restart following power restoration.
The following function block must therefore be called in OB21 and in OB22 with the value "+10".

| STL FB2 | Explanation |
|---|---|
| NAME :ZYKZEIT<br>DECL :ZEIT I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KF | |
| : LW =ZEIT<br>:<br>: T BS 96<br>: BE | Load the scan monitoring<br>time as a multiple of 10 msec.<br>Transfer time to system<br>data word 96. |

## 2.7 Accessories

The following subsections give an overview of important accessories for the S5-115U PLC.

### 2.7.1 Backup Battery

The backup battery maintains the program and data when the S5-115U is switched off. The backup battery has a service life of approximately two years.
After a lengthy storage period lithium batteries develop a passivation coating which is responsible for the "voltage delay effect" (when the battery is loaded, the voltage drops temporarily to under 3 V).
Before using a lithium battery in the power supply module, you should depassivate the battery by loading it for two hours with 100 ohms.

> **Note**
>
> The safety regulations for hazardous materials must be observed when transporting lithium batteries!

### 2.7.2 Memory Submodules

There are three types of memory submodules with different memory capacities available for the S5-115U:

**Table 2-8. Available Memory Submodules**

| Memory Submodule | | Cable Used in CPU | | | | Order Number | Program No. | Organization |
|---|---|---|---|---|---|---|---|---|
| Type | Capacity* | 941 | 942 | 943 | 944 | | | |
| EPROM | 8 Kbytes | ▒ | ▒ | ▒ | ▒ | 6ES5 375-1LA15 | 11 | byte |
| EPROM | 16 Kbytes | ▒ | ▒ | ▒ | ▒ | 6ES5 375-1LA21 | 12 | byte |
| EPROM | 32 Kbytes | | ▒ | ▒ | ▒ | 6ES5 375-1LA41 | 17 | byte |
| EPROM | 64 Kbytes** | | | ▒ | ▒ | 6ES5 375-1LA61 | 122 | byte |
| EPROM | 128 Kbytes*** | | | | ▒ | 6ES5 375-1LA71 | 124/163 | word |
| EEPROM | 8 Kbytes | ▒ | ▒ | ▒ | ▒ | 6ES5 375-0LC31 | 211 | byte |
| EEPROM | 16 Kbytes | ▒ | ▒ | ▒ | ▒ | 6ES5 375-0LC41 | 212 | byte |
| RAM**** | 8 Kbytes | ▒ | ▒ | | | 6ES5 375-0LD11 | --- | byte |
| RAM**** | 16 Kbytes | ▒ | ▒ | | | 6ES5 375-0LD21 | --- | byte |
| RAM**** | 32 Kbytes | | ▒ | | | 6ES5 375-0LD31 | --- | byte |

* 2 Kbyte corresponds to approximately 1000 STEP 5 statements

** Only 48 Kbytes can be used

*** Only 96 Kbytes can be used

** ** It is no longer necessary to use RAM submodules in the CPU 943 and CPU 944 since these CPUs provide the entire memory capacity in the form of internal RAM.

### 2.7.3 Programmers (PG)

Applications:
- entering programs
- testing programs
- monitoring programs.

You can use the following programmers: PG 605U, PG 635, PG 670, PG 675, PG 685, PG 695, PG 710, PG 730, PG 750 and PG 770.

You can use the programmers in either on-line or off-line mode on.

### 2.7.4 Operator Panels (OP)

Applications:
- displaying the current values of internal timers and counters
- entering new setpoints
- displaying program-controlled messages
- displaying input, output, data, and flag areas (OP 396 only)

You can use the following operator panels: OP 393, OP 395 and OP 396
You can use the following monitors: DG 335 and DS 075
(DS 075 only with CPU 943/944 (ASCII driver))

### 2.7.5 Printers (PT)

Applications:
- listing inputs
- listing outputs
- listing programs

You can use the following printers: PT 88, PT 89 and PT 90

CP 525 and CP 523 communications processors, programmers (from PG 605U upward) and the SI 2 interface of the CPU 943/944 (ASCII driver required) can be connected.

## 3 Installation Guidelines


3.1 Mounting Rack . . . . . 3. - 1
3.1.1 Central Controller (CC) . . . . . 3. - 1
3.1.2 Expansion Unit (EU) . . . . . 3. - 8

3.2 Mechanical Installation . . . . . 3. - 13
3.2.1 Installing the Modules . . . . . 3. - 13
3.2.2 Installing Fans . . . . . 3. - 16
3.2.3 Dimension Drawings . . . . . 3. - 17
3.2.4 Cabinet Installation . . . . . 3. - 18
3.2.5 Centralized Configurations . . . . . 3 - 19
3.2.6 Distributed Configuration . . . . . 3 - 20
3.2.7 Other Possible Configurations . . . . . 3 - 28

3.3 Wiring . . . . . 3. - 29
3.3.1 Connecting the PS 951 Power Supply Module . . . . . 3 - 29
3.3.2 Connecting Digital Modules . . . . . 3 - 30
3.3.3 Front Connectors . . . . . 3. - 31
3.3.4 Simulator . . . . . 3. - 32
3.3.5 Connecting the Fan Subassembly . . . . . 3 - 33

3.4 General Configuration . . . . . 3. - 33
3.4.1 Power Supply . . . . . 3. - 33
3.4.2 Electrical Installation with Field Devices . . . . . 3 - 35
3.4.3 Connecting Nonfloating and Floating Modules . . . . . 3 - 40

3.5 Wiring Arrangement, Shielding and Measures against Electromagnetic Interference . . . . . 3 - 42
3.5.1 Running Cables Inside and Outside a Cabinet . . . . . 3 - 42
3.5.2 Running Cables Outside Buildings . . . . . 3 - 43
3.5.3 Equipotential Bonding . . . . . 3. - 44
3.5.4 Shielding Cables . . . . . 3. - 45
3.5.5 Special Measures for Interference-Free Operation . . . . . 3 - 46

## Figures


3-1. Central Controller (Example) . . . . . 3 - 1
3-2. Possible Configurations on Mounting Rack CR 700-0 (6ES5 700 0LA12) . . . 3 - 2
3-3. Possible Configurations on Mounting Rack CR 700-0 (6ES5 700 0LB11) . . . . 3 - 3
3-4. Possible Configurations on Mounting Rack CR 700-1 . . . . . 3 - 4
3-5. Possible Configurations on Mounting Rack CR 700-2 . . . . . 3 - 5
3-6. Possible Configurations on Mounting Rack CR 700-2LB . . . . . 3 - 6
3-7. Possible Configurations on Mounting Rack CR 700-3 . . . . . 3 - 7
3-8. Expansion Unit (Example 1) . . . . . 3 - 8
3-9. Possible Configurations on Mounting Rack ER 701-0 . . . . . 3 - 9
3-10. Possible Configurations on Mounting Rack ER 701-1 . . . . . 3 - 10
3-11. Possible Configurations on Mounting Rack ER 701-2 . . . . . 3 - 11
3-12. Possible Configurations on Mounting Rack ER 701-3 . . . . . 3 - 12
3-13. Installing the Modules . . . . . 3 - 13
3-14. Slot Coding Element . . . . . 3 - 14
3-15. Installation of a Printed Circuit Board into an Adapter Casing (6ES5 941-0LB11) . . . . . 3 - 15
3-16. Installing the Fan Subassembly . . . . . 3 - 16
3-17. Dimension Drawings of Mounting Racks . . . . . 3 - 17
3-18. Dimensions for Installation in a 19-inch Cabinet . . . . . 3 - 18
3-19. Centralized Configuration with the IM 305 and IM 306 Interface Modules . . . . . 3 - 19
3-20. Distributed Configuration with IM 304/IM 314 . . . . . 3 - 22
3-21. Switch and Jumper Settings on the IM 304-3UA1. for Distributed Configurations . . . . . 3 - 23
3-22. Switch and Jumper Settings on the IM 304-3UB1. for Distributed Connection . . . . . 3 - 25
3-23. Jumper Settings on the IM 314 . . . . . 3 - 27
3-24. Power Supply Module PS 951 . . . . . 3 - 29
3-25. Connection to Floating and Nonfloating Modules . . . . . 3 - 30
3-26. Front Connectors, Front View . . . . . 3 - 31
3-27. Installing the Front Connector . . . . . 3 - 32
3-28. Simulators . . . . . 3 - 32
3-29. Fan Subassembly Terminal Assignment . . . . . 3 - 33
3-30. Operating a Programmable Controller with Field Devices on Grounded Supply . . . . . 3 - 37
3-31. Operating a Programmable Controller with Field Devices on Centrally Grounded Supply . . . . . 3 - 38
3-32. Operating a Programmable Controller with Field Devices on Nongrounded Supply . . . . . 3 - 39
3-33. Simplified Representation of an Installation with Nonfloating Modules . . 3 - 40
3-34. Simplified Representation for Installation with Floating Modules . . . . . 3 - 41

## Figures


3-35. Laying Equipotential Bonding Conductor and Signal Cable . . . . . 3 - 44
3-36. Fixing Shielded Cables with Various Types of Cable Clamps . . . . . 3 - 46
3-37. Wiring Coils . . . . . 3 - 46
3-38. Measures for Suppressing Interference from Fluorescent Lamps in the Cabinet . . . . . 3 - 47


## Tables


3-1. Dimension Drawings of Modules . . . . . 3 - 16
3-2. Comparison of the IM 305 and IM 306 Interface Modules . . . . . 3 - 18
3-3. Technical Specifications for Distributed Configuration Interface Modules . . . . . 3 - 19
3-4. Connection of the S5-115U System to other SIMATIC S5 Systems . . . . . 3 - 26
3-5. Front Connector Overview . . . . . 3 - 29
3-6. Rules for Common Running of Lines . . . . . 3 - 42

# 3 Installation Guidelines

Programmable controllers of the S5-115U system consist of a central controller to which one or more expansion units can be connected if necessary. The modules that make up the S5-115U programmable controller (PLC) are mounted on racks.

## 3.1 Mounting Rack

Various mounting racks are available to suit the performance or the degree of expansion the control system is to have. Each mounting rack consists of an aluminum mounting rail for fastening all modules mechanically and one or two backplanes for connecting the modules to each other electrically. The module locations (slots) are numbered in ascending order from left to right.

### 3.1.1 Central Controller (CC)

A central controller has a power supply module (PS), a central processing unit (CPU), and various input/output modules (I/Os). Depending on requirements, digital or analog modules, communications processors (CPs), or intelligent input/output modules can be used. Figure 3-1 shows a basic CC configuration.

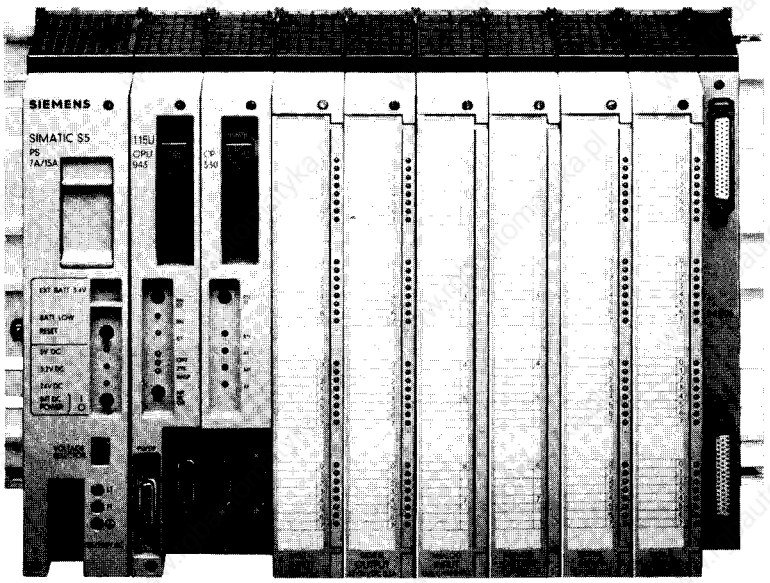



**Figure 3-1. Central Controller (Example)**

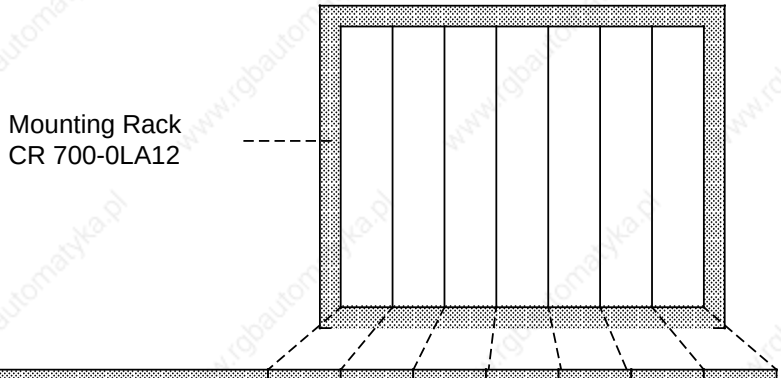
The following five mounting racks (CRs) are available for installing a central controller:

- CR 700-0LA12 and CR 700-0LB11: for central controller 0
- CR 700-1: for central controller 1
- CR 700-2: for central controller 2
- CR 700-3: for central controller 3

They differ in the number of slots and in configuration possibilities. The following pages describe the four mounting rack types for central controllers and the modules that can be mounted on them (connector pin assignment see Appendix C).

**Possible Configurations on Mounting Rack CR 700-0 (6ES5 700-0LA12)**

Use central mounting rack CR 700-0 to install small control systems. It has slots for a power supply module (PS), a central processing unit (CPU), and up to four input/output modules (I/Os). Such a configuration makes up a central controller 0. You can plug in an interface module (IM) to connect expansion units (EUs), and the CP 530 communications processor to connect the SINEC-L1 local area network. You can also mount one intelligent input/output module. Figure 3-2 shows the slots on CR 700-0 and the modules that can be plugged into each slot.




| Slots | PS | CPU | 0 | 1 | 2 | 3 | IM |
|---|---|---|---|---|---|---|---|
| Power Supply Module | ▒ | | | | | | |
| Central Processing Unit | | ▒ | | | | | |
| Digital Module [1] | | | ▒ | ▒ | ▒ | ▒ | |
| Analog Module [1] | | | ▒ | ▒ | ▒ | ▒ | |
| IM 305/IM 306 [2] | | | | | | | ▒ |
| CP 530-7 | | | ▒ | | | | |
| CP 523 | | | ▒ | | | | |
| IP 240/241/242 /243 /244 /260 /261 | | | ▒ | | | | |
| DIMOS Interface Module | | | ▒ | | | | |

▒ Only with adapter casing

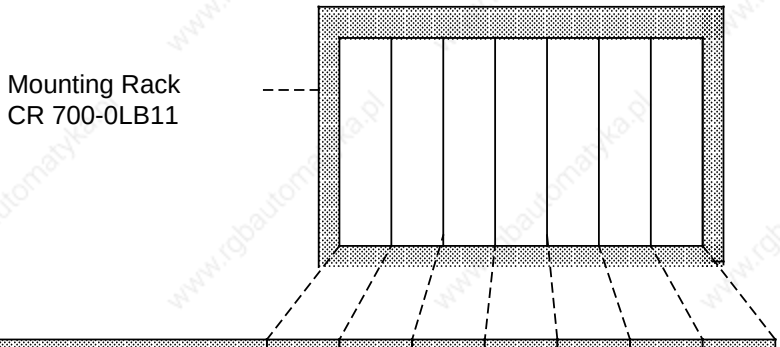
1 Digital/analog modules of ES 902 design (S5-135/155U) can only be used in slot 0!

2 Do not remove the termination connector if neither IM 305 nor IM 306 is plugged in.

**Figure 3-2. Possible Configurations on Mounting Rack CR 700-0 (6ES5 700-0LA12)**

**Possible Configurations on Mounting Rack CR 700-0 (6ES5 700-0LB11)**

You can use adapter casings with two printed-circuit boards in the case of the CR 700-0 (6ES5 700-0BL11) in contrast to the CR 700-0 (6ES5 700-0LA12). There are also slots for a power supply module (PS), a central processing unit (CPU), block type digital/analog modules, intelligent input/output modules and communications processors (CPs).




| Slots | PS | CPU | 0 | 1 | 2 | 3 | IM |
|---|---|---|---|---|---|---|---|
| Power Supply Module [1] | ▒ | | | | | | |
| Central Controller | | ▒ | | | | | |
| Digital Module [2] | | | ▒ | ▒ | ▒ | | |
| Analog Module [2] | | | ▒ | ▒ | ▒ | | |
| CP 513/523/524/525/552-1 | | | ▒ | ▒ | ▒ | | |
| CP 526/527 Basic Board | | | ▒ | ▒ | ▒ | | |
| CP 581/CP 526 Expansion Board | | | | ▒ | | | |
| CP 530-7 | | | | ▒ | ▒ | | |
| CP 5430 | | | ▒ [3] | ▒ | ▒ | | |
| CP 552-2 | | | ▒ [4] | | | | |
| CP 535-3MA12/CP 143-0AB01 | | | ▒ [4] | ▒ | ▒ | | |
| IP 241/245/252 | | | ▒ [4] | ▒ | ▒ | | |
| IP240/242 ... 244/246/247/260/261 | | | ▒ | ▒ | ▒ | | |
| WF 625/725/470 | | | ▒ | ▒ | ▒ | | |
| DIMOS-VISRAM | | | ▒ | ▒ | ▒ | | |
| IM 304/307/308 AS 301/302 | | | | | | ▒ | |
| IM 305/306 [5] | | | | | | | ▒ |
| IM 307 [6] | | | ▒ | ▒ | ▒ | ▒ | |

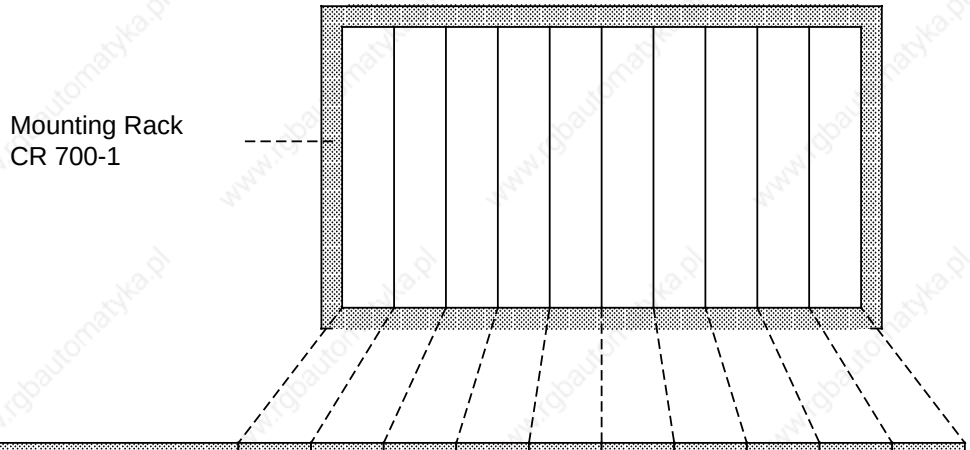
▒ Only with adapter casing

1 Use of the IP 246/247 and the CP 513/524/525/526/527/539/143 is not permissible with a 3 A power supply module (the DSI signal is not generated from the 3 A power supply)
2 Digital/analog modules of ES 902 design (S5-135/155U) can be plugged into slot 0 to 2; block-type modules can be plugged into slot 1 and 2
3 Can only be used in the lefthand slot when operating without fan
4 Slot cannot be used because of double-width module
5 Do not remove the terminating connector if neither the IM 305 nor IM 306 interface module is plugged in
6 Interrupt processing is not possible in slot 3

**Figure 3-3. Possible Configurations on Mounting Rack CR 700-0 (6ES5 700-0LB11)**

**Possible Configurations on Mounting Rack CR 700-1**

Use central mounting rack CR 700-1 to install small or medium-sized control systems. It has slots for a power supply module (PS), a central processing unit (CPU), and up to seven input/output modules (I/Os). Such a configuration makes up a central controller 1. Central controller 1 is upwardly compatible to central controller 0. You can plug in an interface module (IM) to connect expansion units (EUs), and the CP 530 communications processor to connect the SINEC-L1 local area network. You can also mount one intelligent input/output module (IP, WF). Figure 3-4 shows the slots on CR 700-1 and the modules that can be plugged into each slot.




| Slots | PS | CPU | 0 | 1 | 2 | 3 | 4 | 5 | 6 | IM |
|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply Module | ▒ | | | | | | | | | |
| Central Processing Unit | | ▒ | | | | | | | | |
| Digital Modules [1] | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| Analog Modules [2] | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| IM 305 / IM 306 [3] | | | | | | | | | | ▒ |
| CP 530-7 | | | ▒ | | | | | | | |
| CP 523 | | | ▒ | | | | | | | |
| IP 240/241 ... 245/260/261 | | | ▒ | | | | | | | |
| WF 625/725 | | | ▒ | | | | | | | |
| DIMOS Interface Module | | | ▒ | | | | | | | |

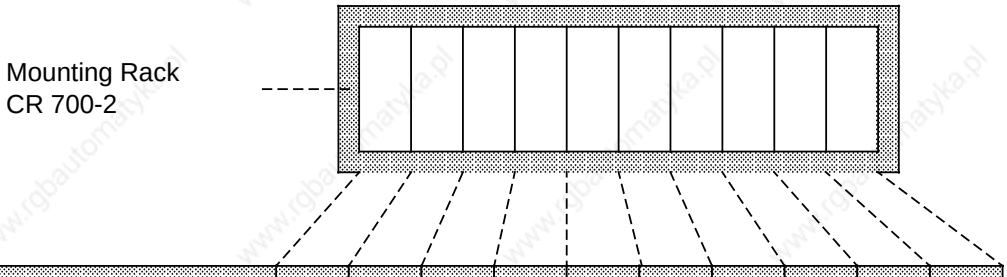
▒ Only with adapter casing

1 Digital modules of ES 902 design (S5-135/155U) can only be plugged into slot 0
2 Plug analog modules into slots 4, 5 and 6 only if an IM 306 is used. Plug the 466 analog input module only into slot 0!
3 Do not remove the termination connector if neither IM 305 nor IM 306 interface module is plugged in.

**Figure 3-4. Possible Configurations on Mounting Rack CR 700-1**

**Possible Configurations on Mounting Rack CR 700-2**

Use central mounting rack CR 700-2 to install large control systems in 19-inch cabinets. It has slots for a power supply module (PS), a central processing unit (CPU) and input/output modules. Such a configuration makes up a central controller 2. You can plug in an interface module (IM) to connect expansion units (EUs). You can also mount intelligent input/output modules and communications processors (CPs). Figure 3-5 shows the slots on CR 700-2 and the modules that can be plugged into each slot.




| Slots | PS | CPU | 0 | 1 | 2 | 3 | 4 | 5 | 6 | IM |
|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply Module [1] | ▒ | | | | | | | | | |
| Central Processing Unit | | ▒ | | | | | | | | |
| Digital Modules [2] | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| Analog Modules [3] | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| IM 305/IM 306 [4] | | | | | | | | | | ▒ |
| AS 301/302/IM 304/308 | | | | | | | | | ▒ | |
| IM 307 [5] | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| CP 143-0AB01/CP 5430 | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| CP 513/523 ... 526/527/ 535/552-1 | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| CP 530-7 | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| IP 240/241 ... 246/247/260/261 | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| IP 252 [6] | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| WF 625/725/470/480 | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| DIMOS Interface Module | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |

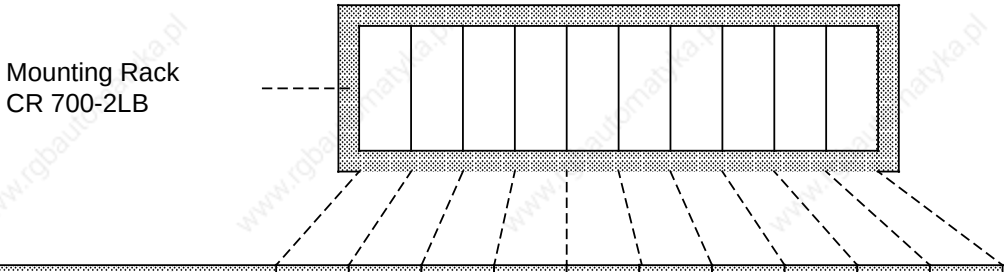
▒ Only with adapter casing

1 Use of the IP 246/247 and the CP 513/524/525/526/527/535/143 is not permissible with a 3 A power supply module. The DSI signal is not generated by the power supply
2 Digital modules of ES 902 design (S5-135/155U) can be plugged into slots 0 to 5
3 Block-type modules can be plugged into slots 0 to 6 and into slots 4, 5 and 6 only when using an IM 306; digital modules of ES 902 design (S5-135/155U) can be plugged into slots 0 to 5
4 Do not remove the termination connector if neither IM 305 nor IM 306 interface module is plugged in
5 Interrupt processing is not possible in slot 6
6 Direct I/O access of the IP 252 with CPU 941, 942 or 943 is only possible in slot 0; CPU 944 only in slots 0, 1, 2; direct I/O access is in general not possible when using the 3 A power supply module (HOLD and HOLDA signals are not available)

**Figure 3-5. Possible Configurations on Mounting Rack CR 700-2**

**Possible Configurations on Mounting Rack CR 700-2LB**

Use central mounting rack CR 700-2LB to install large control systems in 19-inch cabinets. In contrast to the central mounting racks CR 700-0/1 you can use adapter casings with two printed circuit boards here. The CR 700-2LB has slots for a power supply module (PS), a central processing unit (CPU), block-type digital and analog modules, intelligent input/output modules (IPs) and communications processors (CPs). You can plug in an interface module to connect expansion units. Figure 3-6 shows the slots on CR 700-2LB and the modules that can be plugged into each slot.




| Slots | PS | CPU | 0* | | 1* | | 2* | | 3 | 4 | 5 | 6 | IM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply Module [1] | ▩ | | | | | | | | | | | | |
| Central Processing Unit | | ▩ | | | | | | | | | | | |
| Digital Module | | | | | | | | | ▩ | ▩ | ▩ | ▩ | |
| Analog Module | | | | | | | | | ▩ | ▩ 4 | ▩ 4 | ▩ 4 | |
| CP 513/523/524/525/530-3/552-4 | | | ▨ | | ▨ | | ▨ | | ▨ | ▨ | ▨ | | |
| CP 526/527 Basic module | | | ▨ | | ▨ | | ▨ | | ▨ | ▨ | ▨ | | |
| CP 526 Expansion module [2] | | | | ▨ | | ▨ | | | | | | | |
| CP 530-7LA12 | | | | | | | ▩ | | ▩ | ▩ | ▩ | | |
| CP 552-2 | | | ▨ | 3 | ▨ | 3 | | | | | | | |
| IP 241/245/252/CP 535-3MA12 | | | ▨ | 3 | ▨ | 3 | ▨ | | ▨ | ▨ | ▨ | | |
| IP 240/242...244/246/247/260/261 | | | ▨ | | ▨ | | ▨ | | ▨ | ▨ | ▨ | | |
| WF 625/725/470 | | | ▨ | | ▨ | | ▨ | | ▨ | ▨ | ▨ | | |
| DIMOS-VISRAM | | | ▨ | | ▨ | | ▨ | | ▨ | ▨ | ▨ | | |
| IM 305/IM 306 [4] | | | | | | | | | | | | ▨ | |
| AS 301/302/IM 304/307/308 | | | | | | | | | | | | | ▩ |

▨ Only with adapter casing 6ES5 491-0LB11

* Adapter casing 6ES5 491-0LB11 only can be used in these slots.
1 Use of the IP 246/247 and the CP 513/524/525/526/527/535/143 is not permissible with a 3 A power supply module.
2 Can only be used in adapter casing 6ES5 491-0LB11 in conjunction with CP 526 basic module.
3 Slot cannot be used because of double-width module.
4 Can only be addressed with IM 306.

**Figure 3-6. Possible Configurations on Mounting Rack CR 700-2LB**

**Possible Configurations on Mounting Rack CR 700-3**

Use central mounting rack CR 700-3 to install large control systems in 19-inch cabinets. In contrast to the central mounting racks CR 700-0/1/2, you can also use printed circuit boards in an adapter casing here. The CR 700-3 has slots for a power supply module (PS), a central processing unit (CPU), input/output modules, intelligent input/output modules (IPs) and communications processors (CPs). You can plug in an interface module to connect expansion units. A configuration on the CR 700-3 makes up a central controller 3. Figure 3-7 shows the slots on the CR 700-3 and the modules that can be plugged into each slot.

| Slots | PS | CPU | 0 | | 1 | | 2 | | 3 | 4 | 5 | 6 | | IM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply Module [1] | ▒ | | | | | | | | | | | | | |
| Central Processing Unit | | ▒ | | | | | | | | | | | | |
| Digital Modules [2] | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| Analog Modules [3] | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | | |
| CP 513/523 ... 525/552-1 | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | | |
| CP 526 Basic Module/CP 581 [9] | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | | |
| CP 526 Expansion Module | | | | ▒ | | ▒ | | ▒ | | | | | | |
| CP 530-7 | | | | | | | | | ▒ | ▒ | ▒ | | | |
| CP 5430 | | | ▒ | 4 | ▒ | 4 | ▒ | 4 | ▒ | ▒ | ▒ | | | |
| CP 552-2 | | | ▒ | 5 | ▒ | 5 | ▒ | 5 | | | | | | |
| CP 535/CP 143-0AB01 | | | ▒ | 5 | ▒ | 5 | ▒ | 5 | ▒ | ▒ | ▒ | | | |
| IP 241/245/252 [6] | | | ▒ | 5 | ▒ | 5 | ▒ | 5 | ▒ | ▒ | ▒ | | | |
| IP | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | | |
| WF 625/725/470/480 | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | | |
| WF 485 | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | | | | | |
| DIMOS-VISRAM | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | | |
| IM 304/308 AS 301/302 | | | | | | | | | | | | ▒ | ▒ | |
| IM 307 [8] | | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| IM 305/306 [7] | | | | | | | | | | | | | | ▒ |

▒ Only with adapter casing ▒ Only with adapter casing 6ES5 491-0LC11

1 Use of the IP 246/247 and CP 513/524/525/526/527/535/143 is not permissible with a 3 A power supply module (the DSI signal is not generated by the power supply)
2 Digital modules of ES 902 design (S5-135/155U) can be plugged into slots 0 to 5; block-type modules can plugged into slots 3 to 5; digital module 6ES5 434-4UA12 can also be plugged into slot 6.
3 Block-type modules can plugged into slots 3 to 5; block-type modules on slots 4 and 5 can only be addressed with the IM 306; analog modules of ES 902 design (S5-135/155U) can be plugged into slots 0 to 5
4 Can only be used in the lefthand slot when operating without fan
5 Slot cannot be used because of double-width module
6 Direct I/O access with CPU 941, 942 or 943 is only possible in slot 0; with CPU 944 only in slots 0, 1, 2; direct I/O access is in general not possible when using the 3 A power supply module (HOLD and HOLDA signals are not available)
7 Do not remove the termination connector if neither IM 305 nor IM 306 interface module is plugged in
8 Interrupt processing is not possible in slot 6
9 CP 581, 6ES5 581-1ED13 can only be plugged into the left-hand slot.

**Figure 3-7. Possible Configurations on Mounting RackCR 700-3**

### 3.1.2 Expansion Unit (EU)

If the central controller does not have sufficient slots for the whole PLC, it is possible to include one or more expansion units. Depending on the type of connection, there are four subracks available for expansion units:

- ER 701-0 for expansion unit "0" (EU 0)
- ER 701-1 for expansion unit "1" (EU 1)
- ER 701-2 for expansion unit "2" (EU 2)
- ER 701-3 for expansion unit "3" (EU 3)

(see Appendix C for connector pin assignment)

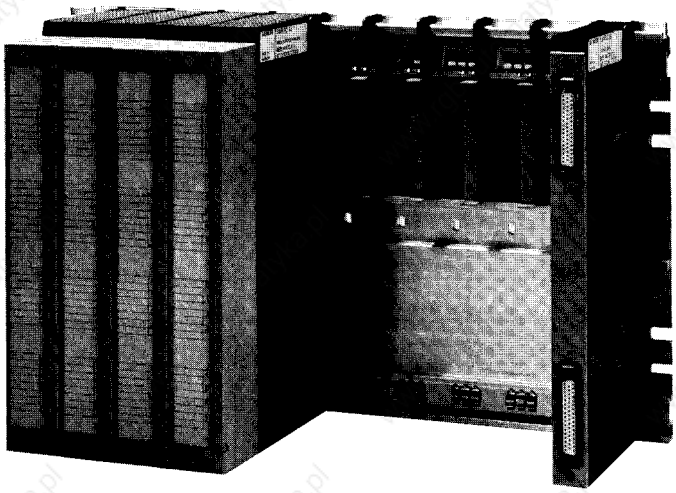


**Figure 3-8. Expansion Unit 1 (Example)**

The following expansion unit interface modules connect EU 1 expansion units to a central controller in centralized configurations (see Section 3.2.5):

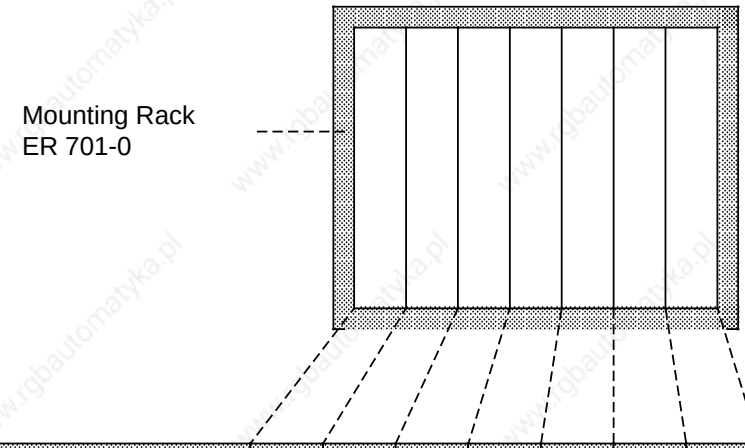
- IM 305
- IM 306

The following interface modules connect expansion units to a central controller in distributed configuration (see Section 3.2.6):

- interface module AS 301/AS 310
- interface module AS 302/AS 311
- interface module IM 304/IM 314
- interface module IM 307/IM 317
- interface module IM 308/IM 318

**Possible Configurations on Mounting Rack ER 701-0**

Use expansion mounting rack ER 701-0 to install expansion unit EU 0. EU 0 is suitable for centralized configurations, i.e. connection to a local central controller of type CC 0, CC 1 or CC 2. The ER 701-0 has six slots for digital and analog input or output modules and one slot for an IM 305 or IM 306 interface module. Interrupt-triggering modules cannot be used. The expansion unit gets its power supply via the EU interface module. You can connect up to three expansion units to one central controller (CC 0/CC 1/CC 2/CC 3) or to one EU 2/3.




| Slots | 0 | 1 | 2 | 3 | 4 | 5 | IM |
|---|---|---|---|---|---|---|---|
| Digital Modules 1 | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| Analog Modules 2 | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| IM 305/IM 306 | | | | | | | ▒ |

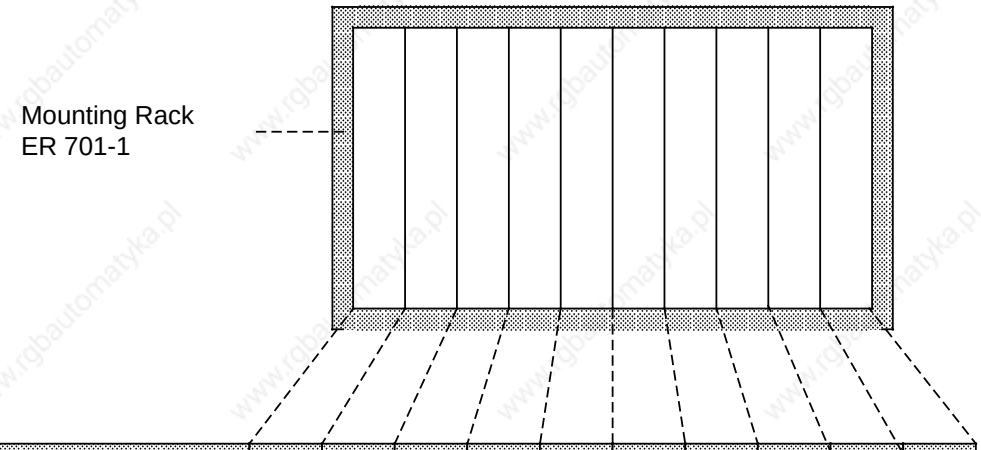
1 The 434-7 input module cannot be plugged into these slots
Digital modules of ES 902 design (S5-135/155U) cannot be plugged in

2 Only if an IM 306 is used.
Analog modules of ES 902 design (S5-135/155U) cannot be plugged in

**Figure 3-9. Possible Configurations on Mounting Rack ER 701-0**

**Possible Configurations on Mounting Rack ER 701-1**

Use expansion mounting rack ER 701-1 to install expansion unit EU 1. EU 1 is suitable for centralized configurations, i.e., connection to a local central controller of type CC 0, CC 1, CC 2 or CC 3. The ER 701-1 has nine slots for digital and analog input or output modules and one slot for an IM 305 or IM 306 expansion unit interface module. Interrupt-triggering modules cannot be used. The expansion unit gets its power supply via the EU interface module. You can connect up to three expansion units to one central controller (CC 0/CC 1/CC 2/CC 3) or to one EU 2/3 expansion unit.




| Slots | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | IM |
|---|---|---|---|---|---|---|---|---|---|---|
| Digital Module [1] | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| Analog Module [2] | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| IM 305/IM 306 | | | | | | | | | | ▒ |

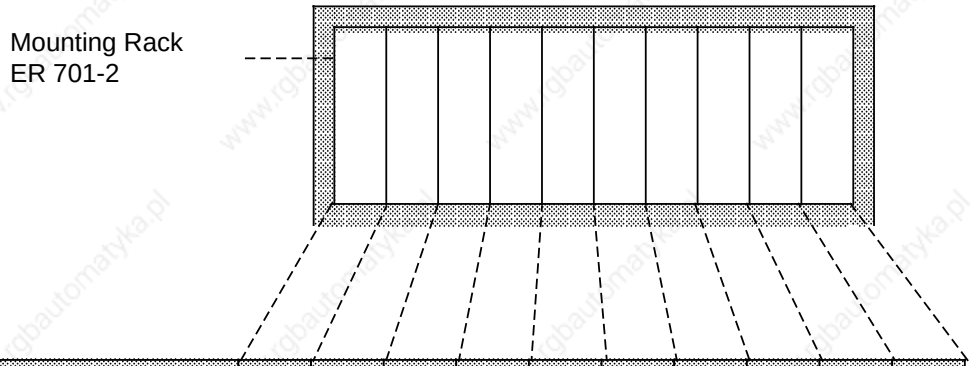
1 Input module 434-7 cannot be plugged into these slots.
Digital modules of ES 902 design (S5-135/155U) cannot be plugged in

2 Only if an IM 306 is used.
Analog modules of ES 902 design (S5-135/155U) cannot be plugged in

**Figure 3-10. Possible Configurations on Mounting Rack ER 701-1**

**Possible Configurations on Mounting Rack ER 701-2**

Use expansion mounting rack ER 701-2 to install expansion unit EU 2. EU 2 is suitable for connection to a centrally configured or remote CC 2/CC 3 central controller. The ER 701-2 has slots for a power supply module (PS), digital and analog input or output modules, one central controller interface module, and one IM 306 expansion unit interface module. The IM 306 lets you connect up to three EU 1 expansion units to one EU 2 expansion unit. Interrupt-triggering modules cannot be used.
Use the IM 310, IM 311, IM 314 and IM 318 interface modules to connect the EU 2 to the S5-135U, S5-150U and S5-155U programmable controllers.




| Slots | PS | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | IM |
|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply Module | ▨ | | | | | | | | | |
| Digital Modules [1] | | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | |
| Analog Modules [2] | | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | ▨ | |
| IM 305/IM 306 | | | | | | | | | | ▨ |
| AS 310/311/IM 314/317/318 | | | | | | | | | ▨ | |
| ÜBW 313 [3] | | | | | | | | ▨ | | |
| CP 523 [3] | | | | | | | | ▨ | | |

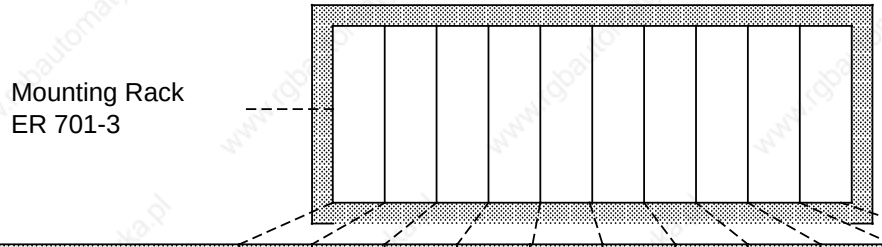
▨ Only with adapter casing

1 The 434-7 input module cannot be plugged into these slots; digital modules of ES 902 design (S5-135/155U) can only be plugged into slot 6
2 Only if an IM 306 is used. Not permissible with the AS 302/311. Analog modules of ES 902 design (S5-135/155U) can only be plugged into slot 6
3 Not permissible with the AS 302/311

**Figure 3-11. Possible Configurations on Mounting Rack ER 701-2**

**Possible Configurations on Mounting Rack ER 701-3**

Use expansion mounting rack ER 701-3 to install expansion unit EU 3. EU 3 is suitable for connection to a centrally configured or remote CC 2/CC 3 central controller. ER 701-3 has slots for a power supply module (PS), digital and analog input or output modules, communications processors and intelligent input/output modules (interrupt-triggering modules can only be used via an IM 307/317), one central controller interface module, and one IM 306 expansion unit interface module. The IM 306 lets you connect up to three EU 1 expansion units to one EU 3 expansion unit.
Use the IM 310, IM 311, IM 314 and 318 central controller interface modules to connect the EU 3 to the S5-135U, S5-150U and S5-155U programmable controllers.




| Slots | PS | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | IM |
|---|---|---|---|---|---|---|---|---|---|---|
| Power Supply Module 1 | ▒ | | | | | | | | | |
| Digital Module 2 | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| Analog Module 3 | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| IM 306 | | | | | | | | | | ▒ |
| IM 314/318 AS 310/311 | | | | | | | | | ▒ | |
| IM 317 | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | |
| CP 513/523...526/ 535/552 4 | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| CP 530-7 | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| CP 4 5430/CP 4 143-0AB01 | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| IP 241 ... 245/260/261 | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| IP 4 240/246/247/252 | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| WF 625/725/470/480 | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |
| ÜBW 313 3 | | ▒ | | | | | | | | |
| DIMOS Interface module 4 | | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | ▒ | | |

▒ Only with adapter casing

1 Use of the IP 246/247 and the CP 513/524/525/526/527/535/143 is not permissible with a 3 A power supply module (the DSI signal is not generated by the power supply)
2 Except the 434-7 input module; digital modules of ES 902 design (S5-135/155U) can be plugged into slot 0 to 6
3 Not permissible with the AS 302/311; analog modules of ES 902 design (S5-135/155U) can be plugged into slot 0 to 6
4 Only when IM 304 is connected to IM 314 and when IM 307 is connected to IM 317

**Figure 3-12. Possible Configurations on Mounting Rack ER 701-3**

## 3.2 Mechanical Installation

Fasten all modules on the appropriate mounting racks. You can install the mounting racks in cabinets with dimensions in inches or millimeters. Figure 3-13 shows the prescribed method for mounting. You can also fasten the racks to surfaces that are at an angle of up to 15° from a vertical surface. Block-type modules are mounted directly on the rack. Place printed circuit boards in double-height Eurocard format in adapter casings.

### 3.2.1 Installing the Modules

Install block-type modules according to the following procedure:

- Remove the protective caps from the socket connectors on the backplane.
- Hook the top of the module into place between the two guides on the top of the mounting rack.
- Swing the module back until it engages with the socket connectors on the backplane.
- Fasten the screws at the top and bottom of the module.

**Figure 3-13. Installing the Modules**

If the modules are subjected to mechanical vibration, they should be installed as close together as possible.

> **Warning**
>
> Plug in or remove modules only when the power supply has been turned off.

### Mechanical Slot Coding

On the back of each module, with the exception of the power supply and central processing unit, is a slot coding element in the form of a two-part plastic cube. This coding element ensures that when one module is replaced, only another module of the same type will be plugged in in its place.

The coding element consists of two parts, one like a lock and one like a key. The two parts fit together in a defined position. This position is specific to each type of module. When you install the module, the back of the coding element is inserted into the mounting rack. When you swing the module out, the key-shaped part of the element stays in the mounting rack and the lock-shaped part stays on the module.

Now you can install only this particular module or an identical one in this slot. If you want to install a different module, you have to remove the coding element from the mounting rack.

You can also work without slot coding. To do this, you must pull the coding element off the module before you swing the module into place for the first time.

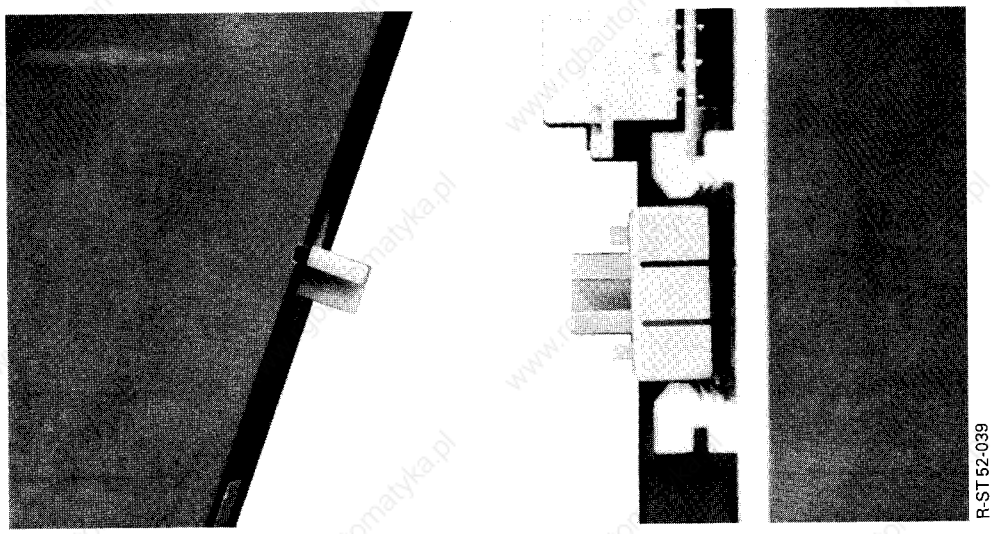



**Figure 3-14. Slot Coding Element**

### Adapter Casing

Use an adapter casing (6ES5 491-0LB11 or 6ES5 491-0LC11) to fasten printed circuit boards in double-height Eurocard format to a mounting rack as you would fasten block type modules.

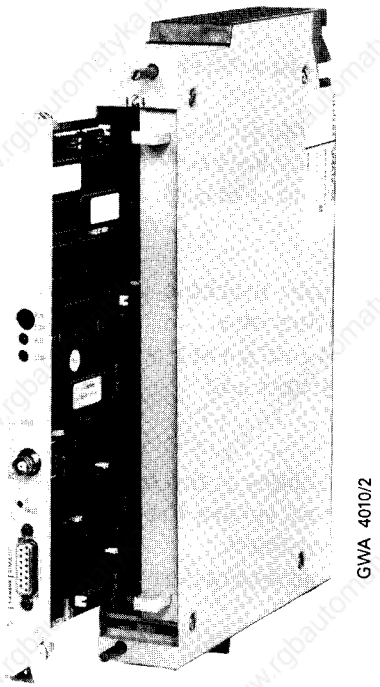



**Figure 3-15. Installation of a Printed Circuit Board into an Adapter Casing (6ES5 491-0LB11)**

Hang the completed unit on the mounting rack and fasten the screws at the top and bottom of the adapter casing.

Push the printed circuit board into the casing along the guide tracks.
Lock the module into place with the eccentric locking collar at the top of the casing.

If an opening remains on the front after the module has been inserted, cover it with a blanking plate.

**Note**

A fan is required for adapter casings with two printed circuit boards.

## 3.2.2 Installing Fans

Install a fan subassembly (6ES5 981-0AA11/21) under the following conditions:

- the power supply modules carry a load of more than 7 A
- the controller uses modules with a high power consumption, e.g., certain communications processors and intelligent input/output modules (see Chapter 15 "Technical Specifications").

The fan subassembly has two fans, a dust filter, and fan monitors with floating changeover contact.
You need a set of installation parts to mount the fan subassemby. The set consists of two installation brackets and a cable duct. The brackets support the fan subassembly and the cable duct. The cable duct enables you to run the field cables off neatly to the side.

Install the fan subassembly as follows:

① Use screws to fasten the installation brackets onto the uprights of the cabinet or on the mounting surface under the mounting rack.
② The guide tracks on the brackets should be at the bottom. Hook the fan subassembly onto the guide tracks of the installation brackets and
③ push it back.
④ Push the fan subassembly up
⑤ and latch it into place with the two slides at the top of the installation brackets.
⑥ If the machine is subject to vibration, secure the fan subassembly to the installation brackets with screws (M 4 x 20 screws with washers).
⑦ Hook the cable duct into the installation brackets.

Special features of the fan subassmbly and installation parts enable you to do the following:

- use the cable duct without the fan subassembly
- install or remove the fan subassembly even when the cable duct is hooked on
- screw the fan subassembly to the installation brackets through the cable duct
- replace the filters while the unit is in operation (see Appendix B).

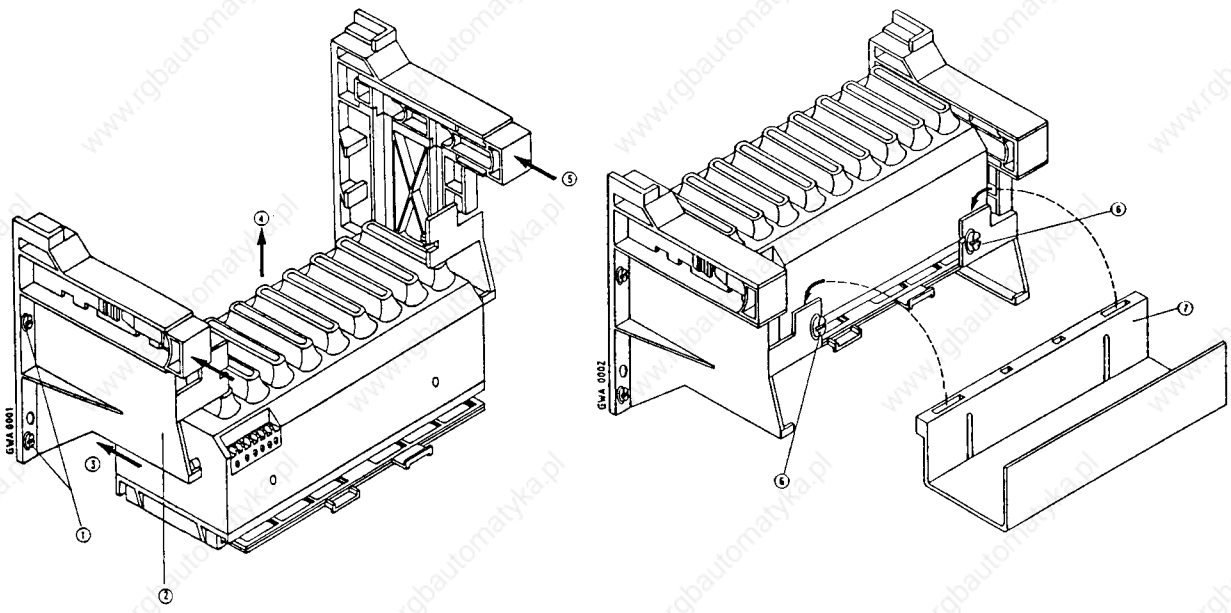
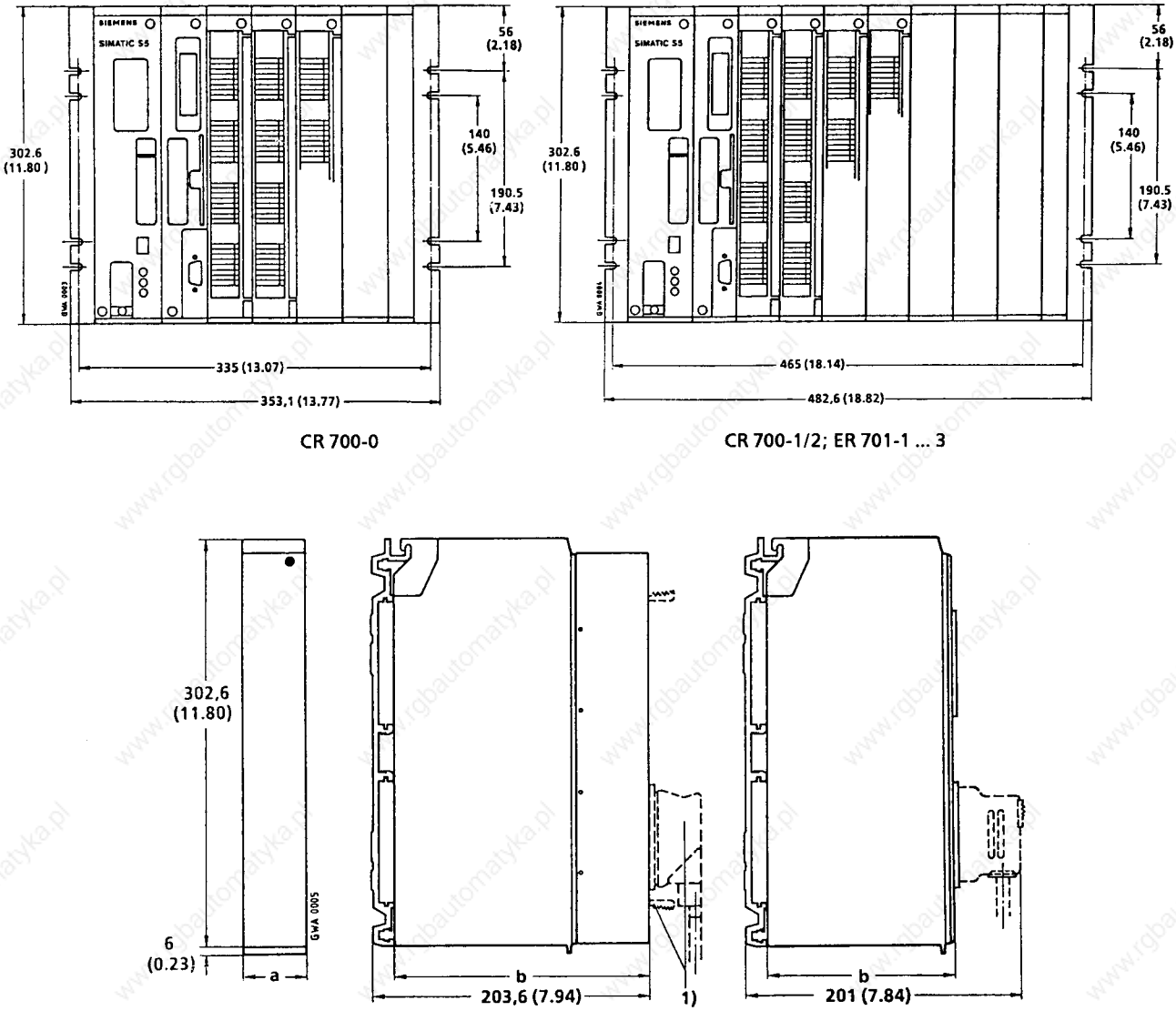



**Figure 3-16. Installing the Fan Subassembly**

## 3.2.3 Dimension Drawings




CR 700-0

CR 700-1/2; ER 701-1 ... 3

1 Control panel and plug connectors (e.g., when an adapter casing is used) extend beyond the front (e.g., CP 525).

**Figure 3-17. Dimension Drawings of Mounting Racks**

**Table 3-1. Dimension Drawings of Modules**

| | a mm (in.) | b mm (in.) | Mechanical Slot Coding |
|---|---|---|---|
| Power Supply Module | 65 (2.54) | 187 (7.29) | none |
| Central Processing Unit | 43 (1.68) | 187 (7.29) | none |
| Digital and Analog Modules | | | built in |
| Adapter Casing | | | |
| Interface Module | 25 (0.98) | 133 (5.19) | none |

## 3.2.4 Cabinet Installation

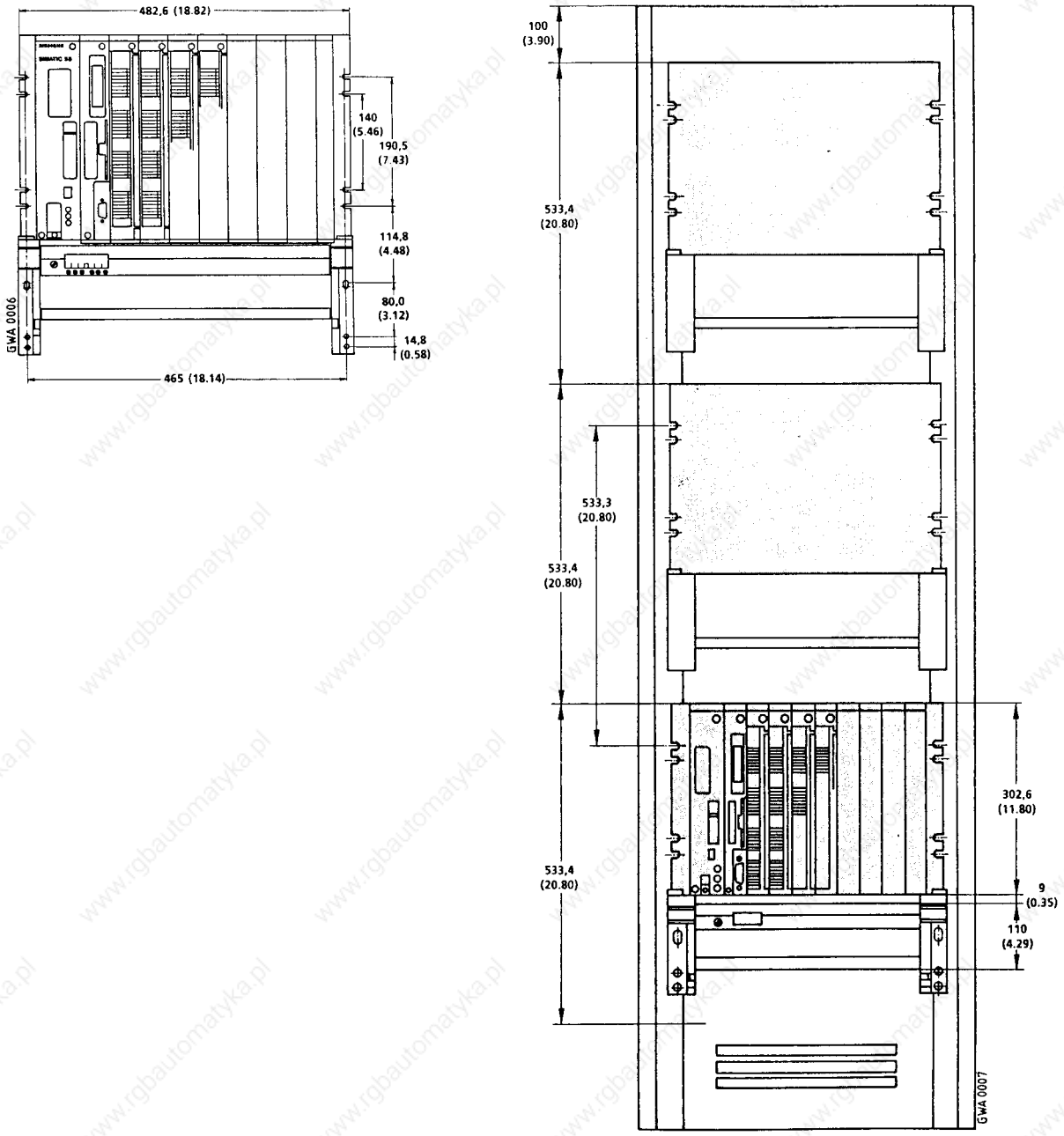



**Figure 3-18. Dimensions for Installation in a 19-Inch Cabinet**

### Caution

The 533.4 mm spacing must be maintained even if a fan is not used.

### 3.2.5 Centralized Configurations

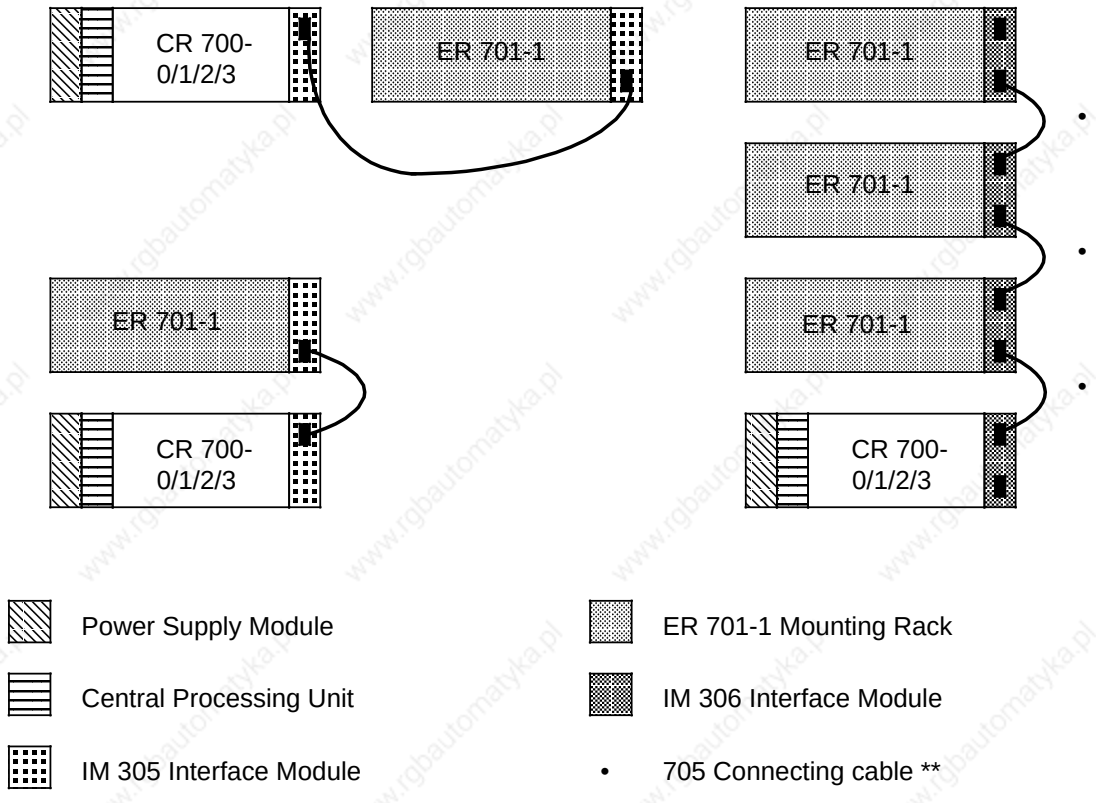
A central controller (CC 0, CC 1, CC 2 or CC 3) connected via short connecting cables to as many as three expansion units makes up a centralized configuration. Use only the IM 305 or IM 306 interface modules to connect an ER 701-1 mounting rack.
For centralized configuration with the IM 305, please note the following points:

- You can use fixed slot addressing only (see Chapter 5).
- The 0.5 m (1.6 ft.) connecting cable is not long enough to connect the EU under the CC. Use an IM 306 interface module or the IM 305 version with a longer cable for such an arrangement.

**Table 3-2. Comparison of the IM 305 and IM 306 Interface Modules**

| | Configuration with IM 305 | Configuration with IM 306 |
|---|---|---|
| Number of EUs (maximum) | 1 | 3 |
| Total cable length | 0.5 or 1.5 m (1.6 or 4.8 ft.) | maximum 2.5 m (8.2 ft.) |
| Slot addressing | fixed (for CC and EU) | variable (for CC and EU) |
| Current supplied to EUs (maximum) | 1 A | 2 A * |

***** The EU with the most current supplied should be as close to the CC as possible.




****** You can also order a 1.25 m (4.1 ft.) 705 connecting cable (Order No. 6ES5 705-0BB20) or a 2.5 m (6.7 ft.) 705 connecting cable (Order No. 6SE5 705-0BC50), and use them to mount two EUs next to each other.

**Figure 3-19. Centralized Configuration with the IM 305 and IM 306 Interface Modules**

### 3.2.6 Distributed Configuration

A central controller connected to expansion units installed over a maximum distance of 3000 m (9800 ft.) makes up a distributed configuration. The interface module used determines the distance and the number of EUs that can be connected.

Distributed configurations using the following are not described here:

- AS 301/AS 310
- AS 302/AS 311
- IM 307/IM 317
- IM 308/IM 318.

You can order these interface modules with a separate description.

Please note the following points concerning distributed configuration versions:

- Each ER 701-2 or ER 701-3 expansion rack requires a PS 951 power supply module and an IM 306 interface module for addressing input/output modules (Exception: ET 100/ET 200).
- If the expansion units have their own power supply, please note the following:
  - When switching on:
    Switch on the power supplies of the expansion units first and only then the power supply of the central controller.
  - If you switch on the power supplies of the central controllers and the expansion units at the same time, you must program a restart delay.
- See Section 3.5 (Shielding)!
- If you use digital input modules on the ER 701-2 or ER 701-3, it is recommended that you use modules with revision level "2" (or higher).

**Table 3-3. Technical Specifications of the Interface Modules for Distributed Configurations**

| | Number of Expansion Units (Max.) | Total Cable Length (Max.) | Power Consumption at 5 V |
|---|---|---|---|
| **AS 301** | 4 | 200 m | 0.8 A |
| **AS 310** | | | 0.7 A |
| **AS 302** | 3 | 1000 m | 2.0 A |
| **AS 311** | | | 1.5 A |
| **IM 304** | 8 | 600 m | 1.2 A |
| **IM 314** | | | 0.85 A |
| **IM 307** | 14* | 50 to 3000 m | 1.0 A |
| **IM 317** | | | 1.0 A |
| **IM 308-3UA** | 63 | 3000 m | 0.5 A |
| **IM 318-3UA or IM 318-8MA** | | | 0.3 A<br>0.35 A at 24 V |
| **IM 308-3UB** | 124 | 23.8 km | 0.6 A |
| **ET 200-Slave Stations** | | | (see ET 200 Manual) |

* Number of EUs depends on the length of the fiber optic cable used and the ready delay time of the individual modules

**Connection with IM 304/IM 314 Interface Modules**

Plug the IM 304 interface module into a CR 700-2/-3-0LB central rack to connect as many as four EUs per interface to the CC. Plug an IM 314 into each ER 701-2 or ER 701-3 expansion rack. Connect the interface modules with the 6ES5 721-.... connecting cable as shown in Figure 3-20.

Connections with the IM 304/IM 314 interface modules have the following special features:

- Using the IM 304/IM 314 symmetrical interface modules, you can connect EUs on ER 701-2 or ER 701-3 expansion racks with full address bus to CCs of the S5-115U, S5-135U, S5-150U, and S5-150S and S5-155U systems.
- Connection to EU 183, EU 185 and EU 186 is possible.
- You can use the extended address set for the programmable controllers mentioned above (see the IM 304/IM 314 manual).
- Always insert a 6ES5 760-1AA11 termination connector in the receptacle for the lower front connector (X4) on the last IM 314.
- The potential difference between CC and EU must not exceed 7 V. An equipotential bonding conductor should therefore be provided!

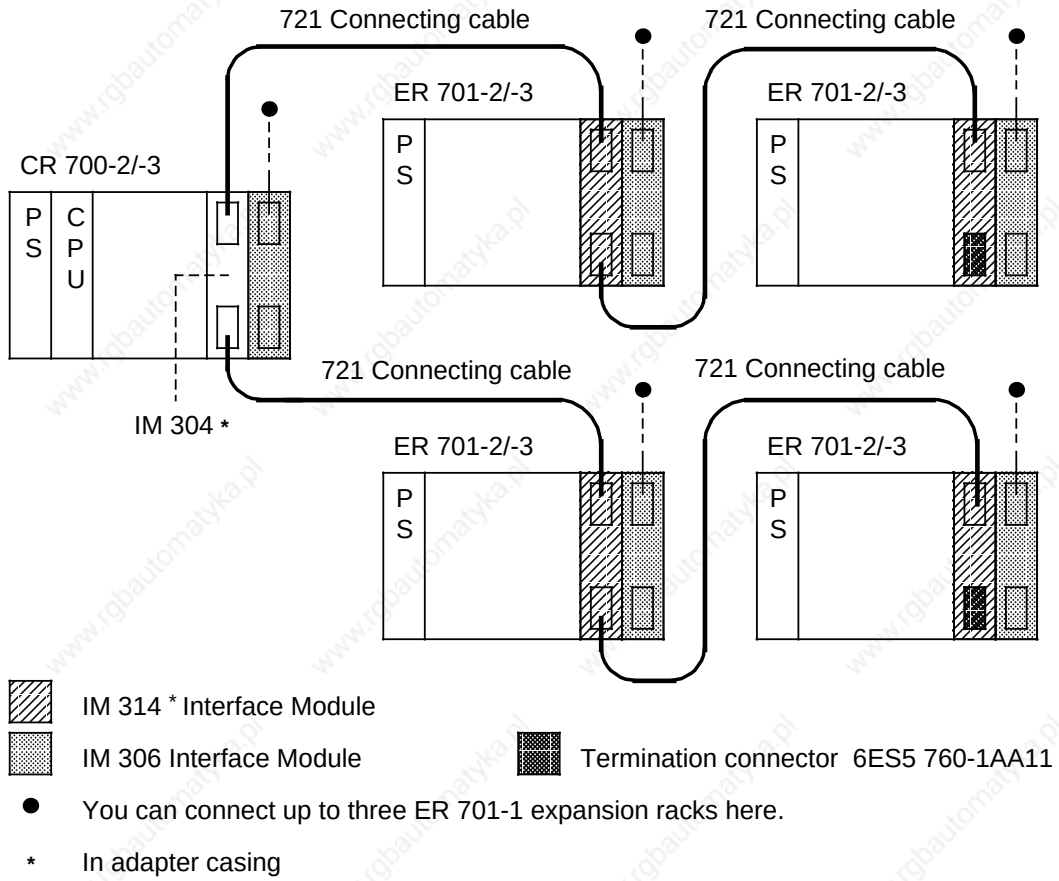



**Figure 3- 20. Distributed Configuration with IM 304/IM 314**

The following is a description of the switch and jumper settings for the IM 304-3U**A**1. and for the IM 304-3U**B**1.

**Switch and Jumper Settings on the IM 304-3UA1. for Distributed Connection**

Figure 3-21 shows the position of the switches and jumpers on the IM 304 module. If you use the IM 304 interface module for distributed configuration, please set the jumpers as shown on jumper block X11.
All switches must be in the "ON" position on block S3.

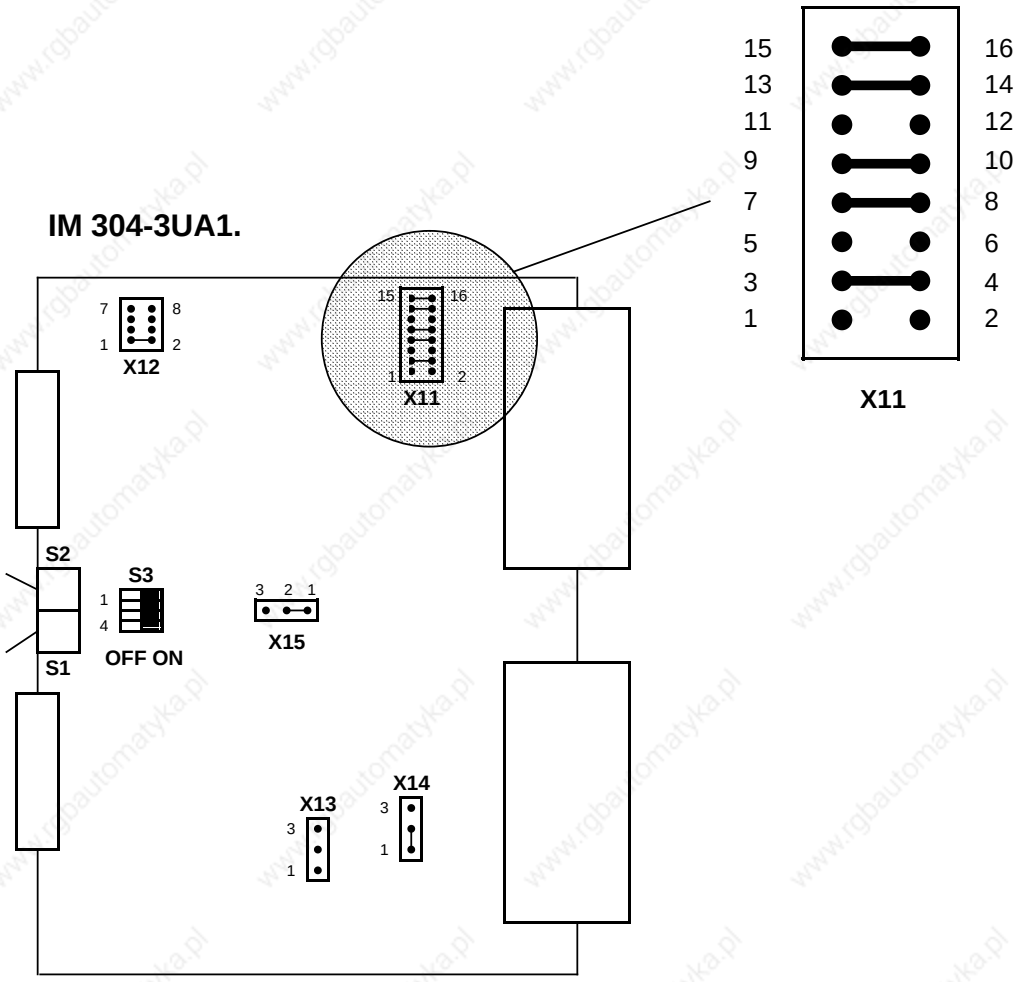
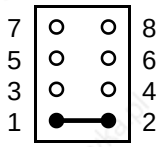



**Figure 3-21. Switch and Jumper Settings on the IM 304-3UA1. for Distributed Configurations**

- Switches S1 and S2 must be in the "ON" position when the relevant interface is in use.

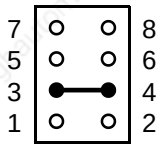
- Set the jumpers on X12 to adapt the cable length for distributed connection.
  When you set the jumpers on X12, use the longest link connected to interface X3 or X4 to determine the setting.
  If you use IPs and CPs on the EU, you must set the longest total length, regardless of the length of the connection line.

X12



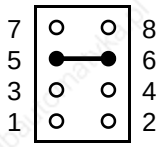

The maximum total length of the distributed connection is 100 m (330 ft.) per interface.

X12



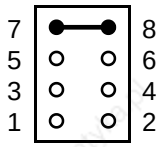

Total length between 100 m (330 ft.) and 250 m (820 ft.)

X12




Total length between 250 m (820 ft.) and 450 m (1500 ft.)

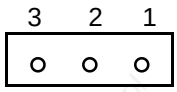
X12



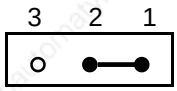

Total length between 450 m (1500 ft.) and 600 m (2000 ft.)

- Set the jumpers on X14 and X15 as follows for the IM 304-3UA1/IM 314 distributed connection:

X14

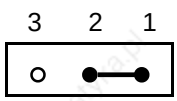



The PEU signal (I/Os not ready) is not evaluated in the ISTACK.

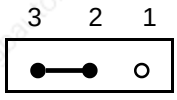



The PEU signal (I/Os not ready) is evaluated in the ISTACK.
Note: When power is turned on, cold restart (RN-ST-RN) is also necessary.

X15




The PEU signal is output in the ISTACK if one interface reports "Not ready".




The PEU signal is output in the ISTACK if both interfaces report "Not ready".

**Switch and Jumper Settings on the IM 304-3UB1. for Distributed Connection**

Figure 3-22 shows the positions of the switches and jumpers on the IM 304-3UB1. module. All switches on switch block S3 must be in the ON position.

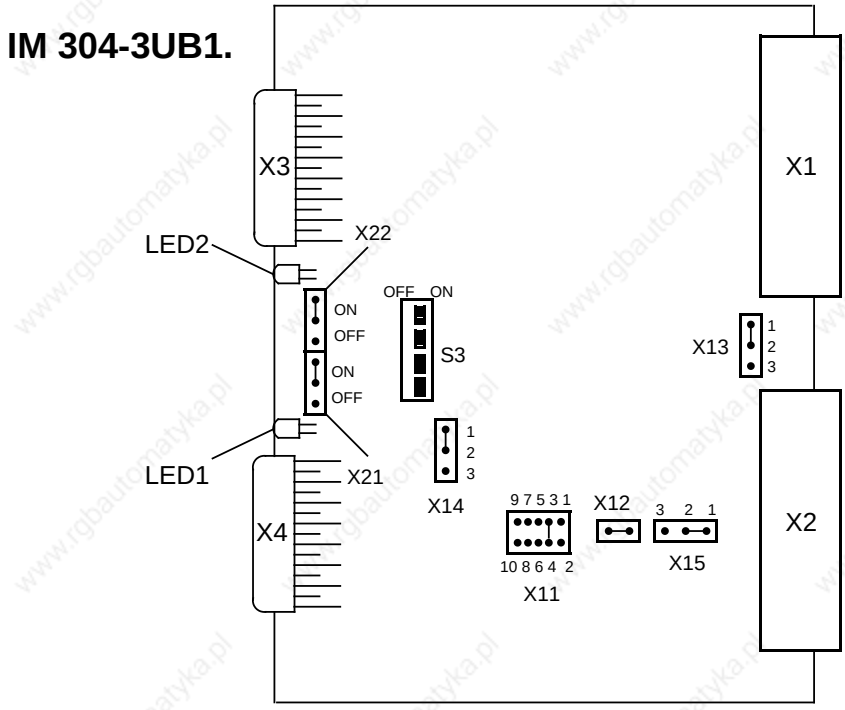



**Figure 3-22. Switch and Jumper Settings on the IM 304-3UB1. for Distributed Connection**

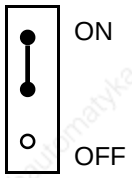
In Figure 3-22, the IM 304 has been set for distributed connection.

- Permissible cable length up to 100 m (330 ft.) (X11)
- PEU signal (I/Os not ready) is located at Pin b18 of the X2 base connector, (setting at X15)
- The PEU signal is generated by the IM 304 if ***at least one*** interface reports "Not ready" (X14)
- An EU is connected to both interfaces (X21 and X22).

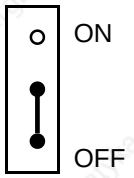
You can change the setting at jumpers X21, X22 as well as at X11, X14 and X15.

- You can switch the interfaces on or off with jumpers X21 and X22.

X21
or
X22

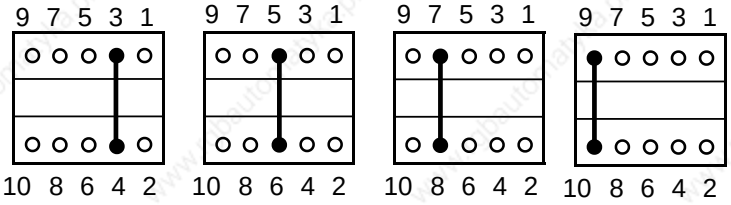



Interface is switched on



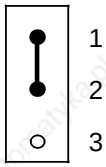

Interface is switched off
(no EU connected)

- Use jumper X11 to set the total length of the 721 connecting cables of one interface up to the last EU. The decisive factor for setting jumper X11 is the interface with the longest connection line.
  If you use IPs and CPs on the EU, you must set the longest cable length!

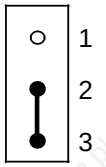
| | **Jumper X11** | | | |
|---|---|---|---|---|
| Jumper location |  | | | |
| Cable length | Up to 100 m (330 ft.) | 100 m (330 ft.) to 250 m (820 ft.) | 250 m (820 ft.) to 450 m (1500 ft.) | 450 m (1500 ft.) to 600 m (2000 ft.) |

- Jumpers X14 and X15 can be set as follows for the IM 304/314 distributed connection:

X14

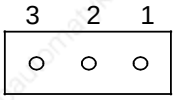



PEU signal is generated if ***at least one*** interface reports "Not ready".

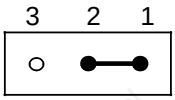



PEU signal is generated only if ***both*** interfaces report "Not ready".

X15




PEU signal is ***not*** evaluated.




PEU signal is evaluated.
Note: When power is turned on in the EU or in the CC, manual cold restart (RN-ST-RN) is also necessary.

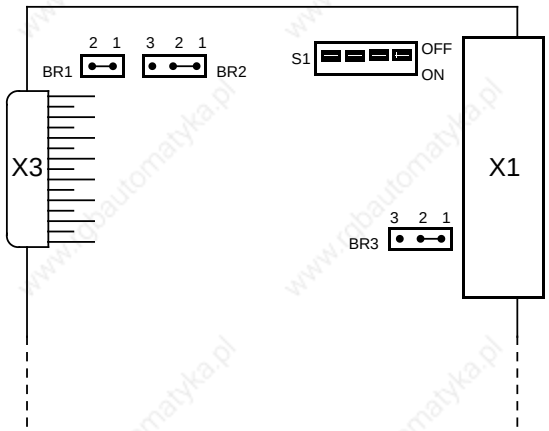
**Note**

If the PEU signal is not evaluated, provision must be made in the restart routine to make sure that the EU is ready for operation before the CC or that the process images are updated in OB1.

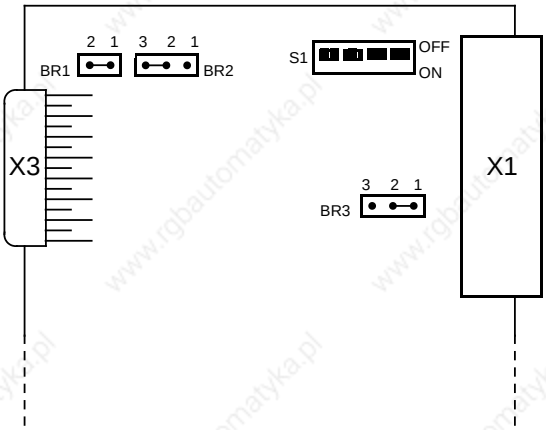
**Switch and Jumper Settings on the IM 314 Interface Module for Distributed Connection**

Jumpers BR1 to BR3 must be set as follows depending on the EU used:

Using the IM 314 in the ER 701-2. ER 701-3 (S5-115U)




Using the IM 314 in the EU 185U and EU 186U




Using the IM 314 in the EU 183U

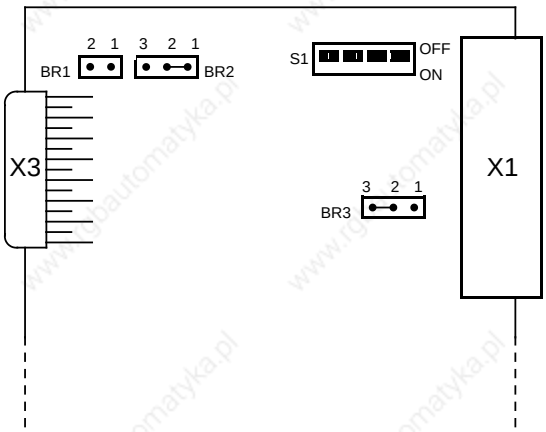



**Figure 3-23. Jumper Settings on the IM 314**

If you use the 313 watchdog module in the EU, you must switch off the PESP (memory I/O select) monitoring facility on the watchdog module.

### 3.2.7 Other Possible Configurations

Central controllers and expansion units of the S5-115U system can also be connected to CCs and EUs of other SIMATIC S5 systems. Table 3-4 shows the possible configurations.

**Table 3-4. Connection of the S5-115U System to other SIMATIC S5 Systems**

| Configuration | Central Controller | Central Controller Interface Module | Expansion Unit | Expansion Unit Interface Module | Connecting cable |
|---|---|---|---|---|---|
| Centralized up to 2.5 m (8 ft.) | 110S<br>130A, 150A | 6ES5 300-5LA11 | EG1 (ER 701-1)<br>or<br>EG2 (ER 701-2 without PS) | 6ES5 306-7LA11 | 705 |
| | 130K, 130W<br>135U, 150K<br>150S, 150U,<br>155U* | 6ES5 300-5LB11 | | | |
| Distributed up to 200 m (650 ft.) | 130A, 150A | 6ES5 301-5AA13 | EG2 (ER 701-2)<br>EG3 (ER 701-3) | 6ES5 310-3AB11 | 721 |
| | 115U<br>130K, 130W<br>135U, 150K<br>150S, 150U,155U | 6ES5 301-3AB13 | | | |
| Distributed up to 600 m (2000 ft.) serial | 135U<br>150S, 150U,155U | 6ES5 304-3UA11 | | 6ES5 314-3UA11 | 721 |
| Distributed up to 1000 m (3800 ft.) serial | 130A,<br>150A | 6ES5 302-5AA11 | | 6ES5 311-3KA11 | 723 |
| | 110S/B<br>130K/W<br>135U<br>150K/S/U **<br>155U | 6ES5 302-3KA11 | | | |

* No word-orientated I/O access (L PW; T PW) possible

** This connection is possible only if a cold restart is prevented by the "STP" statement in OB22

## 3.3 Wiring

The backplane on the mounting rack establishes the electrical connection between all modules.

Make the following additional wiring connections:

- The PS 951 power supply module to the power line
- The sensors and actuators to the digital or analog modules. Connect the sensors and actuators to a front connector that plugs into the contact pins on the front of each module. You can connect the signal lines to the front connector before or after you plug it into the module. The connection diagram of each module is on the inside of the front door. Perforated label strips are included with each input and output module. Use these strips to note the addresses of the individual channels on the module. Slip the strips along with their protective transparent covers into the guides on the front door.
  Chapter 10 "Analog Value Processing" describes how transducers are connected up to analog input modules and the feedback modules of the analog output modules.

Sections 3.3.1 through 3.3.6 explain how to connect individual modules.

Please consult the appropriate operator's guide or manual for information on wiring the intelligent input/output modules and communications processors.

### 3.3.1 Connecting the PS 951 Power Supply Module

Connect the PS 951 power supply module as follows:

Set the voltage selector switch to the appropriate voltage (only in the case of AC modules).

Connect the power cable to terminals L1, N and ⏚.

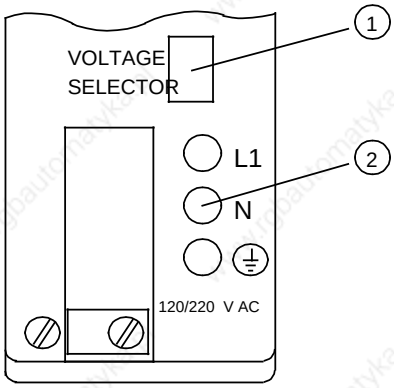



**Figure 3-24. Power Supply Module PS 951**

### 3.3.2 Connecting Digital Modules

Digital modules are available in nonfloating and floating versions. For the nonfloating modules, the reference voltage of the external process signals ($M_{ext}$) has to be connected to the internal reference voltage ($M_{int}$, i.e., PE) (see Figure 3-25). For floating modules, an optocoupler separates the external voltages from the internal ones.

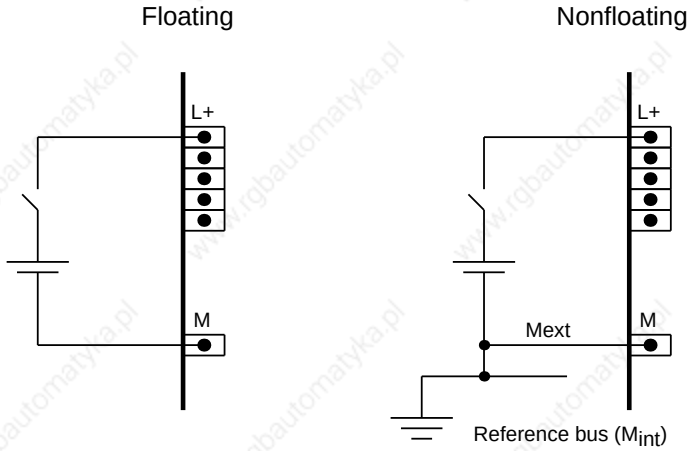



**Figure 3-25. Connection to Floating and Nonfloating Modules**

**Note**

See Chapter 6 "Addressing/Address Assignment" for information on address assignment in the case of digital modules.

### 3.3.3 Front Connectors

Table 3-5 shows the available front connectors. Figure 3-26 shows a front view of these connectors.

**Table 3-5. Front Connector Overview**

| Order No. | Terminals per Front Connector | Connection Method | Wire Cross Section per Terminal * |
|---|---|---|---|
| 6ES5 490-7LB11 | 24 | Screw connection (SIGUT) | 1 x 2.5 mm² to 2 x 1.0 mm² ** |
| 6ES5 490-7LB21 | 46 | Screw connection (Box terminal) | 1 x 2.5 mm² or 2 x 1.5 mm² *** |
| 6ES5 490-7LA11 | 46 | Crimp-snap-in (mini-spring-contact) | several lines from 0.5 to 1.5 mm² total cross section |

* When plug-in jumpers are used, the conductor cross sections are reduced.
** with end sleeves: 0.75 to 1.5 mm² (18 to 15 AWG)
*** with end sleeves: 0.5 to 1.5 mm² (20 to 15 AWG)

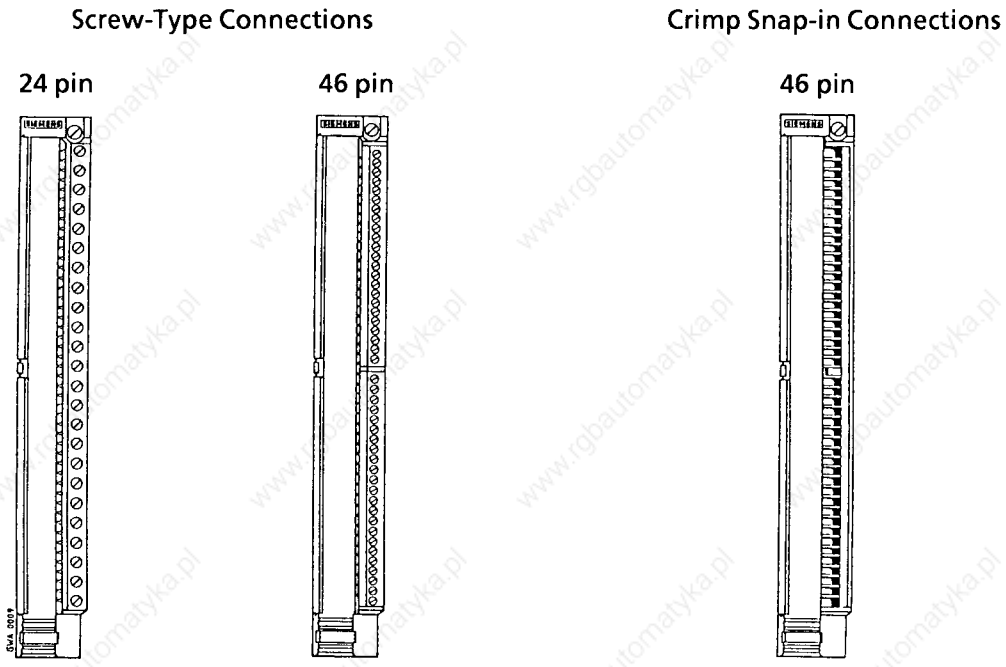


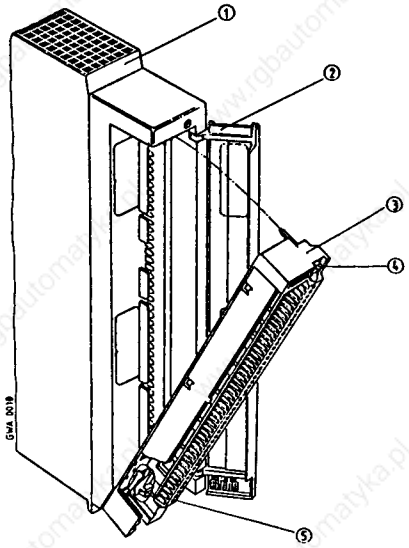

**Figure 3-26. Front Connectors, Front View**

The connectors have openings at the bottom for standard strain-relief clamps.

### Installing the Front Connector

Install the front connector as follows (see Figure 3-27):

1. Open the front door of the module.
2. Hook the front connector in the pivot at the bottom of the module.
3. Swing the front connector up and in until it engages with the module.
4. Tighten the screw at the top of the front connector to secure it.




① Module

② Front door is open

③ Front connector is pushed back

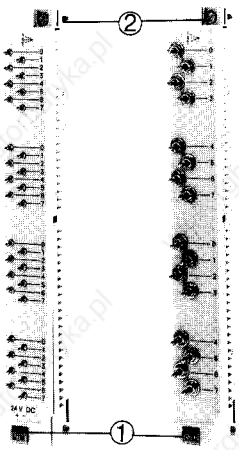
④ Fastening screw

⑤ Pivot

**Figure 3-27. Installing the Front Connector**

## 3.3.4 Simulator

You can use an appropriate simulator instead of a front connector. Use the toggle switches on the front of this device to simulate input signals (see Figure 3-28). A simulator needs an external power supply.




① Screw-type terminals for supply voltage

② Fastening screw

**Figure 3-28. Simulators**

### 3.3.5 Connecting the Fan Subassembly

Figure 3-29 shows the wiring necessary to operate a fan subassembly.

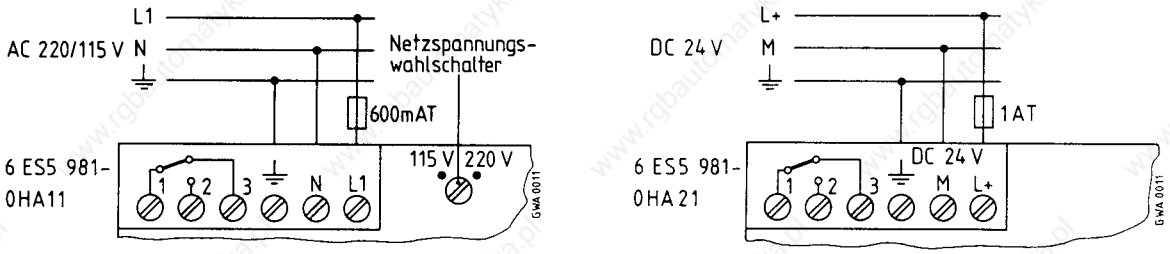



**Figure 3-29. Fan Subassembly Terminal Assignment**

A floating changeover contact gives a fault signal via terminals 1, 2 and 3 if the fan fails.
The diagram in Figure 3-29 shows the switch positions in the case of a fault! Under normal operating conditions, the contacts 1 - 2 are closed and the contacts 1 - 3 open.

## 3.4 General Configuration

The following chapter explains the electrical installation of the S5-115U.

### 3.4.1 Power Supply

A completely installed controller consists of the following separate electrical circuits:
- the control circuit for the central controllers and expansion units
- the load circuit for the sensors and actuators

**The Control Circuit:**

The control circuit supplies the CPU, the I/O bus, the programmer interface, and the control circuits of the input/output modules. The PS 951 power supply module generates the necessary 5 V DC, 5.2 V DC, and 24 V DC operating voltages from the 24 V DC, 120 V AC, or 230 V AC line voltages.
When planning the power supply for the central controllers and expansion units, make sure that the total current consumption of the modules used does not exceed the nominal current of the power supply used. For this reason, we offer the PS 951 power supply module with two ratings, 5 V/3 A and 5 V/7 A (for fan operation, up to 15 A) (see Chapter 2).

When using the different PS 951 power supply modules, you must note the following:

- For the 6ES5 951-7ND41 floating module, the input voltage must be a functional extra-low voltage in accordance with VDE 0100 or a comparable standard. Otherwise, the PE terminal must be conncted to the protective ground wire.
- For the 6ES5 951-7NB21/7ND41/7ND51 power supply modules, there is no galvanic isolation between the 24 V side and the 5 V side whose reference potential is permanently connected to the mounting rack.
- The use of the following modules is ***not*** permissible due to the missing DSI signal in the case of the 3 A power supplies with the order numbers 6ES5 951-7LB14/7NB13:
  - IP 246/247
  - CP 513/526/527/535/580/581/143.
- The CP 524/524 must not be used with 3 A power supply modules since their power consumption is too high.
- Magnetic voltage stabilizers must ***not*** be connected direct on the input side of power supply modules!
  If you use magnetic voltage stabilizers in parallel network branches, you must expect overvoltages to occur as a result of mutual interference. These voltage peaks can destroy the power supply module! If such a case arises, please consult the department responsible.
- The power supply modules 6ES5 951-7LD21/7ND41/7ND51 have 2 back-up batteries. If one of the batteries is discharged, the corresponding signal lamp lights up and the 2nd battery automatically takes over the back-up function.
- You must observe the following for external back-up in the case of the power supply modules with 2 batteries:
  - If you connect an external back-up battery **without** inserting a new battery in the power supply module, the "BATT LOW" LEDs continue to flash.
  - Execute a RESET on the power supply module after connection of the external back-up battery. You thus reset the battery low signal. The "BATT LOW" LEDs, however, continue to flash after the reset.
- You must observe the following for external back-up in the case of the power supply modules with 1 battery:
  - Execute a RESET on the power supply module after connection of the external back-up battery. You thus reset the battery low signal.

**The Load Circuit:**

For monitoring reasons, you should use the same power supply for control and load circuits. For the 24 V DC power supply, a Siemens load power supply unit of the 6EV13 series is recommended (see Catalog ET1).

When connecting other load power supply units, please note the following:

- The output voltage of the internal monitoring circuit of the PC is not detected. Consequently, the load voltage must be monitored externally.
- The output voltage of the power supply unit should not exceed 30 V under partial load conditions.
  Destruction of the modules cannot be ruled out in the case of higher voltages.

### 3.4.2 Electrical Installation with Field Devices

The following figures each show an example circuit for connecting control power supply and load power supply. They also show the grounding concept for operation from the following:

- Grounded supplies
- Centrally grounded supplies
- Nongrounded supplies.

Please note the following when installing your controller. The text contains reference numbers which you can find in Figures 3.30 to 3.32.

**Master switch and short-circuit protection**

- You must provide a master switch to DIN VDE 0113, Part 1, or a disconnecting device to DIN VDE 0100, Part 460, for the programmable controller, sensors and actuators.
  These devices are not required in the case of subsystems where the relevant device has been provided at a higher level.
- You can provide the circuits for sensors and actuators with short-circuit protection and/or overload protection in groups. According to DIN VDE 0100, Part 725, single-pole short-circuit protection is required in the case of grounded secondary side and all-pole protection is required in all other cases.
- For nonfloating input and output modules, connect terminal M of the load power supply unit with the PE ground conductor of the control circuit's PS 951 power supply module.

**Load power supply**

- For 24 V DC load circuits, you require a load power supply unit with safe electrical isolation.
- You require a back-up capacitor (rating: 200µF per 1 A load current) for nonstabilized load power supply units. Connect the capacitor in parallel to the output terminals of the load power supply.
- For controllers with more than five electromagnetic operating coils, galvanic isolation by a transformer is required by DIN VDE 0113, Part 1; it is recommended by DIN VDE 0100, Part 725 .
- For nonfloating input and output modules, connect terminal M of the load power supply unit with the PE ground conductor of the control circuit's PS 951 power supply module.

**Grounding**

- You should ground load circuits where possible . Provide a removable connection to the protective conductor on the load power supply unit (terminal L or M) or at the isolating transformer in secondary circuit.
- To protect against stray noise, use copper conductors of at least 10 $mm^2$ cross section to ground the mounting racks by the shortest possible route.

**Warning**

You must provide insulation monitoring devices for nongrounded power supply modules

- If hazardous plant conditions could arise from double-line-to-ground faults or double fault to frame faults.
- If no safe (electrical) isolation is provided.
- If circuits are operated with voltages > 120 V DC.
- If circuits are operated with voltages > 50 V AC.

- The mounting racks of the S5-115U must be connected to the protective conductor. This grounds the reference potential of the controller.
  Nongrounded operation of S5-115U controllers is only permissible if all the circuits are operated with functional extra-low voltage. In this case, connect the mounting rack or DIN rail over an RC network with the protective conductor.

**Operating a programmable controller with field devices on grounded supply**

Operation from grounded power supplies offers the best protection against interference.

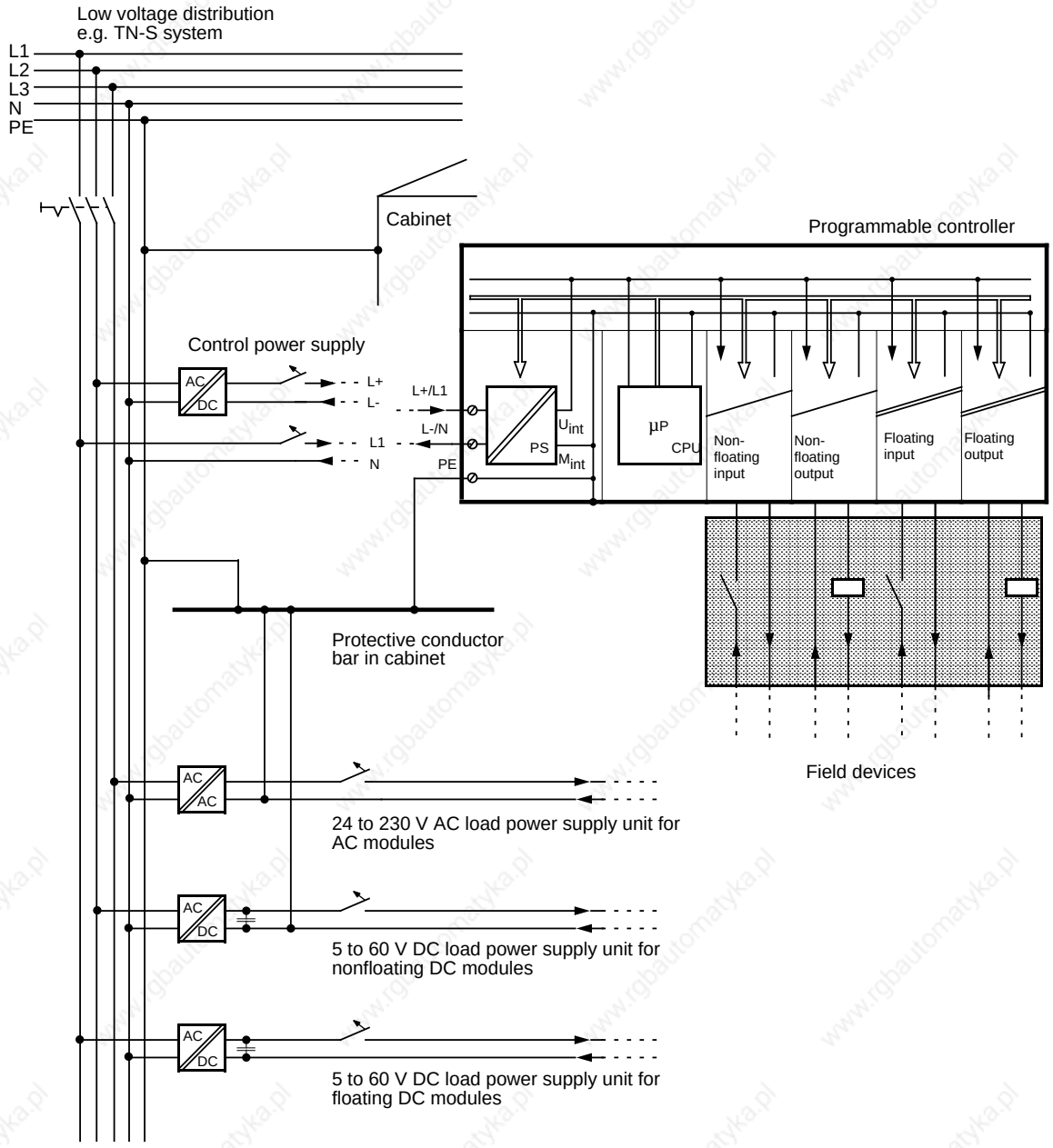



**Figure 3-30. Operating a Programmable Controller with Field Devices on Grounded Supply**

**Operating a programmable controller with field devices on a centrally grounded supply**

In plants with their own transformers or generators, the PLC is connected to the central grounding point. A removable connection must be provided for measuring ground faults.

Installation of the PLC must be such that there is insulation between the cabinet potential and the protective conductor potential. In order to maintain the insulation, all connected devices must be ***grounded capacitively or they must be nongrounded***. For this reason, programmers must be supplied only via an isolating transformer.

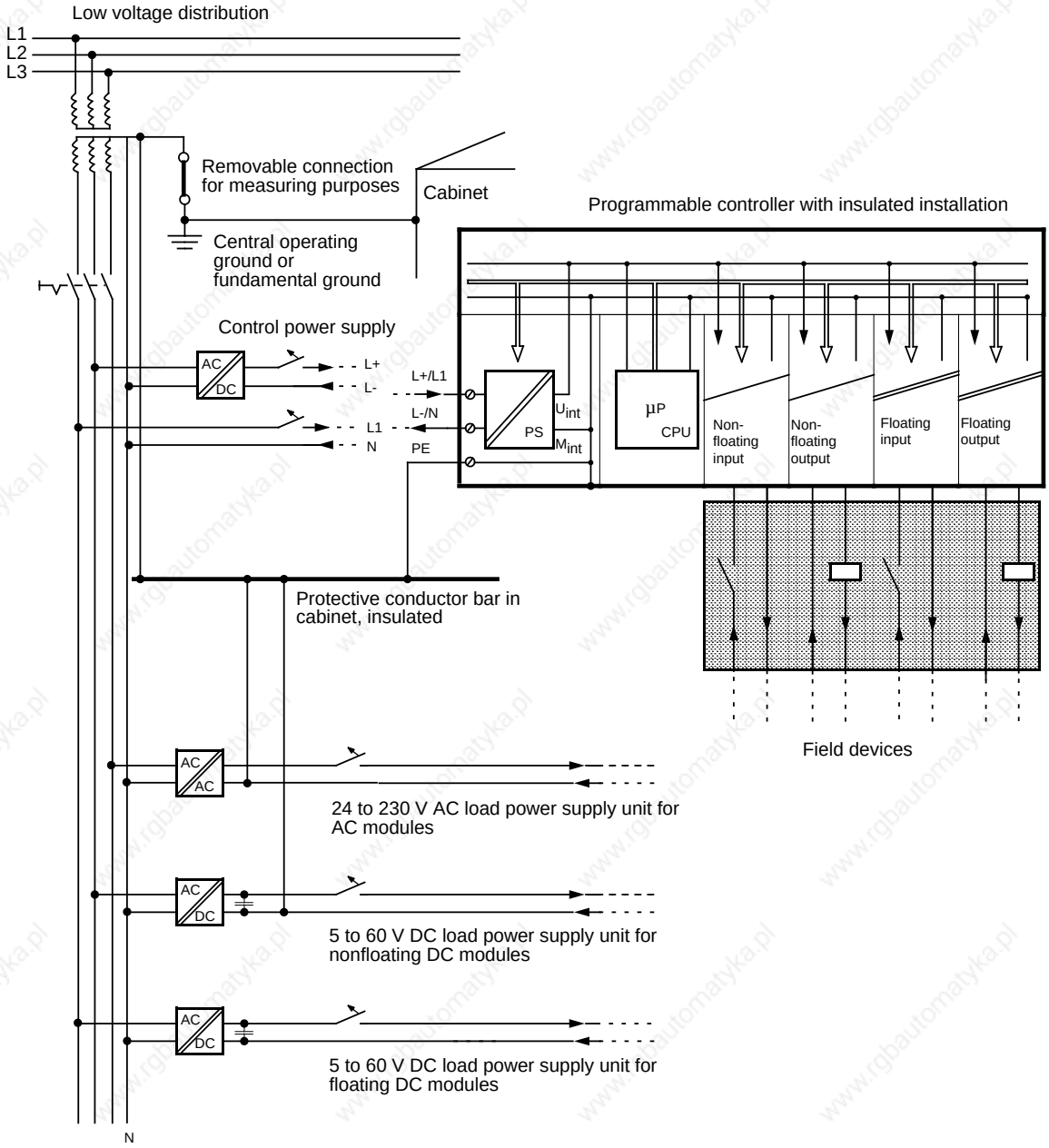



**Figure 3-31. Operating a Programmable Controller with Field Devices on Centrally Grounded Supply**

**Operating a Programmable Controller with Field Devices on Non-grounded Supply**

Neither the outer conductor nor the neutral are connected to the protective conductor in the case of nongrounded supplies. Operation of the PLC with nonfloating power supply modules is ***not*** permissible.
Please note the following when connecting power supply modules:
In networks with 3 x 230 V, you can connect the power supply module direct to two outer conductors (see Figure 3.32).
In networks with 3 x 400 V, connection between the outer conductor and the neutral conductor is not permissible (unacceptably high voltage in the case of ground fault). Use intermediate transformers in these networks.

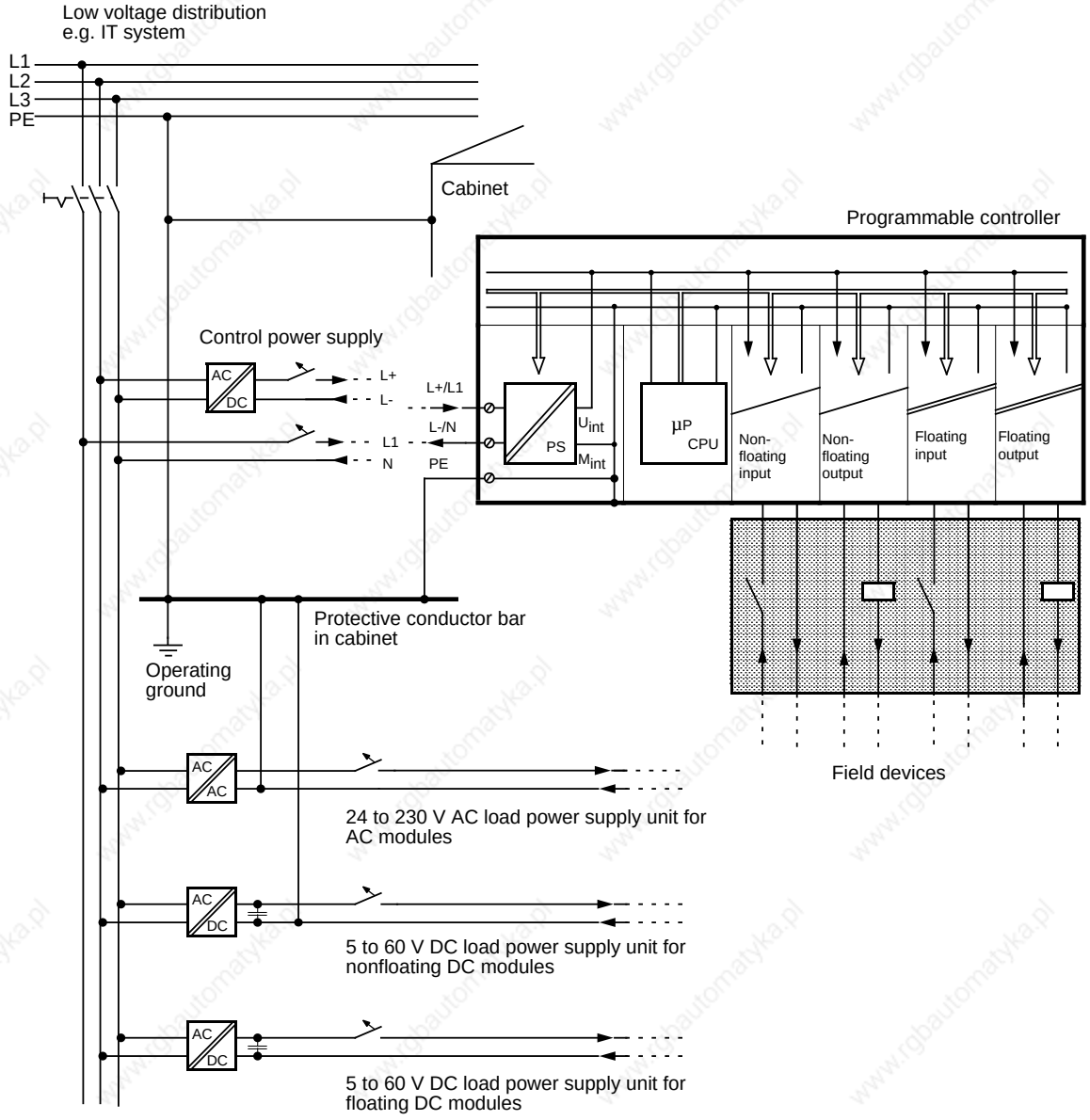



**Figure 3-32. Operating a Programmable Controller with Field Devices on Nongrounded Supply**

## 3.4.3 Connecting Nonfloating and Floating Modules

The following sections show the special features involved in installations with nonfloating and floating modules.

**Installation with nonfloating modules**

In installations with nonfloating modules, the reference potential of the control circuit ($M_{internal}$) and the load circuits ($M_{external}$) are galvanically isolated.

The reference potential of the control circuit ($M_{internal}$) is at the PE terminal or ⏚ and must be connected to the reference potential of the load circuit via a line to be run externally.

Figure 3.33 shows a simplified representation of an installation with nonfloating modules. The installation is independent of the grounding concept. The connections for the grounding measures are therefore ***not*** shown:

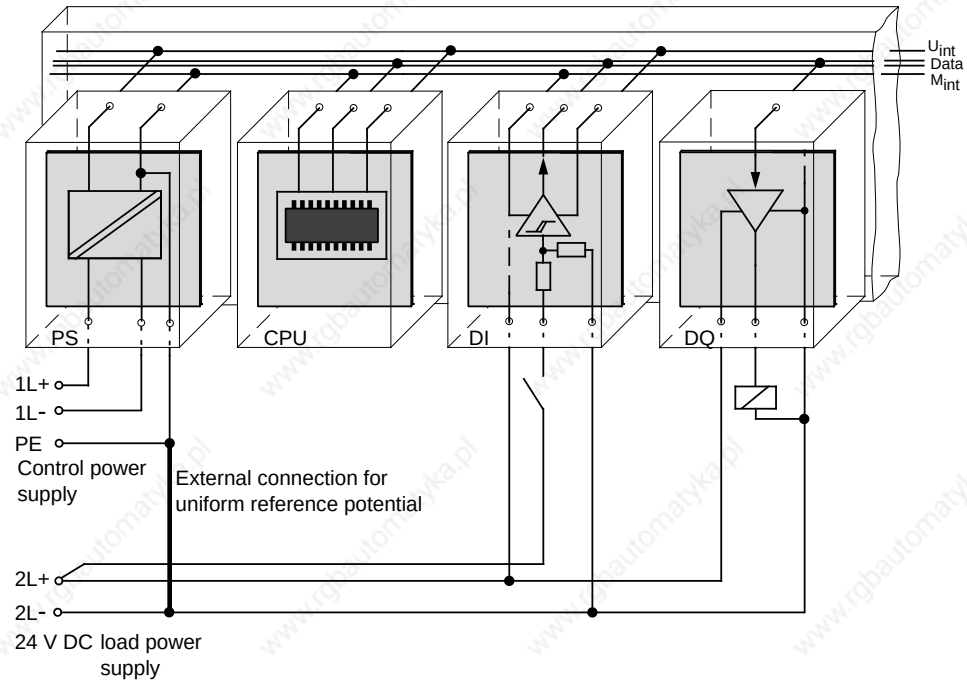



**Figure 3-33. Simplified Representation of an Installation with Nonfloating Modules**

Voltage drop on line must not exceed 1 V, otherwise the reference potentials will shift and result in module failures.

**Note**

It is imperative that you connect the reference potential of the load power supply unit with the L-terminal of the module in the case of 24 V DC DQ modules. If this connection is missing (e.g. wirebreak), a current of typically 15 mA can flow at the outputs. This output current can be sufficient to ensure that

- Energized contactors do not drop out
  and
- High-resistance loads (e.g. miniature relays) can be driven.

**Installation with floating modules**

Control circuit and load circuit are galvanically isolated in the case of floating modules.

Installation with floating modules is necessary in the following cases:

- All AC load circuits
  and
- Non-connectable DC load circuits.
  The reasons for this are, e.g. different reference potentials of the sensors or the grounding of the plus poles of a battery, ...

Figure 3-34 shows the simplified representation of an installation with floating modules. The installation is independent of the grounding concept. The connections for grounding measures are therefore ***not*** shown.

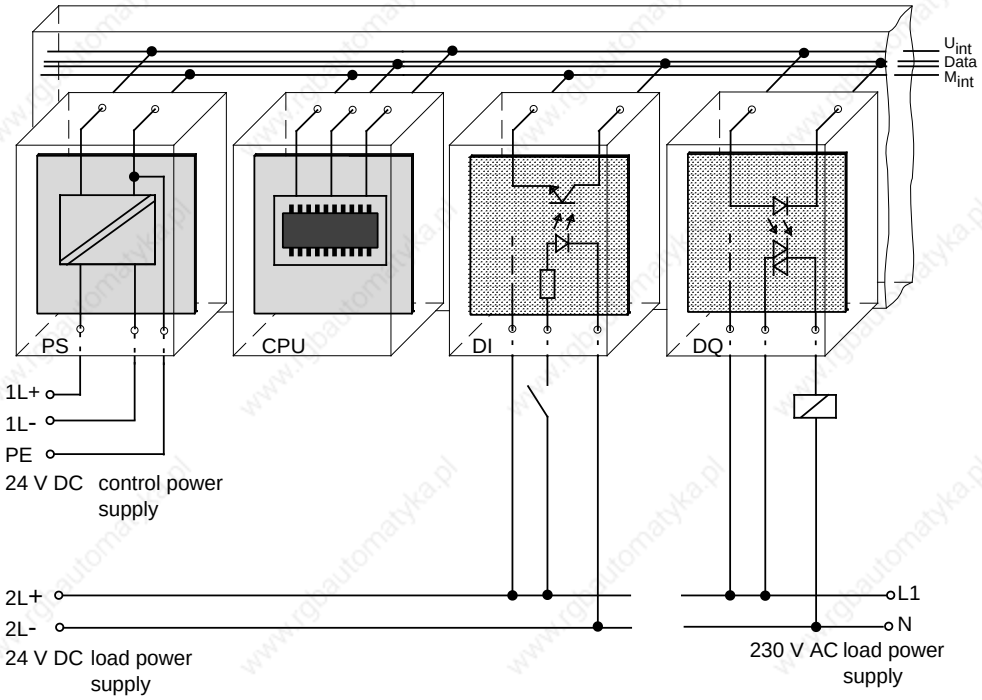



**Figure 3-34. Simplified Representation for Installation with Floating Modules**

## 3.5 Wiring Arrangement, Shielding and Measures against Electromagnetic Interference

This section describes the wiring arrangements for bus cables, signal cables, and power supply cables that guarantee the **e**lectro**m**agnetic **c**ompatibility (EMC) of your installation.

### 3.5.1 Running Cables Inside and Outside a Cabinet

Dividing the lines into the following groups and running the groups separately will help you to achieve electromagnetic compatibility (EMC).

Group A: Shielded bus and data lines (for programmer, OP, printer, SINEC L1, PROFIBUS, Industrial Ethernet, etc.)
Shielded analog lines
Unshielded lines for DC voltage 60 V
Unshielded lines for AC voltage 25 V
Coaxial lines for monitors

Group B: Unshielded lines for DC voltage > 60 V and 400 V
Unshielded lines for AC voltage > 25 V and 400 V

Group C: Unshielded lines for AC and DC voltages > 400 V

You can use the following table to see the conditions which apply to the running of the various combinations of line groups.

**Table 3-6. Rules for Common Running of Lines**

| | Group A | Group B | Group C |
|---|---|---|---|
| **Group A** | | | |
| **Group B** | | | |
| **Group C** | | | |

Legend for table:

Lines can be run in common bundles or cable ducts.
Lines must be run in separate bundles or cable ducts (without minimum distance).
Inside cabinets, lines must be run in separate bundles or cable ducts and outside cabinets but inside buildings, lines must be run on separate cable trays with a gap of a least of 10 cm between lines.

### 3.5.2 Running Cables Outside Buildings

Run lines outside buildings where possible in metal cable supports. Connect the abutting surfaces of the cable supports galvanically with each other and ground the cable supports.

When you run cables outdoors, you must observe the regulations governing lightning protection and grounding. Note the general guidelines:

**Lightning Protection**

If cables and lines for SIMATIC S5 devices are to be run outside buildings, you must take measures to ensure internal and external lightning protection.

Outside buildings run your cables
either

- In metal conduits grounded at both ends
  or
- In steel-reinforced concrete cable channels

Protect signal lines from overvoltage by using:

- Varistors
  or
- Lightning arresters filled with inert gas

Install these protective elements at the point where the cable enters the building.

> **Note**
>
> Lightning protection measures always require an individual assessment of the entire system. If you have any questions, please consult your local Siemens office or any company specializing in lightning protection.

**Grounding**

Make certain that you have sufficient equipotential bonding between the devices.

### 3.5.3 Equipotential Bonding

Potential differences may occur between separate sections of the system if
- Programmable controllers and I/Os are connected via non-floating interface modules or
- Cables are shielded at both ends but grounded via different sections of the system.

Potential differences may be caused, for instance, by differences in the system input voltage. These differences must be reduced by means of equipotential bonding conductors to ensure proper functioning of the electronic components installed.

Note the following for equipotential bonding:

- A low impedance of the equipotential bonding conductor makes equipotential bonding more efficient.
- If any shielded signal cables connected to the ground electrode/protective ground conductor at both ends are laid between the system sections concerned, the impedance of the additional equipotential bonding conductor must not exceed 10 % of the shield impedance.
- The cross-section of the equipotential bonding conductor must be matched to the maximum compensating currents. The following cross-sections are recommendable:
  - 16 mm² copper wire for equipotential bonding line up to 200 m (656.2 ft).
  - 25 mm² copper wire for equipotential bonding line over 200 m (656.2 ft).
- Use equipotential bonding conductors made of copper or zinc-plated steel. Equipotential bonding conductors are to be connected to the ground electrode/protective ground conductor via a large contact area and to be protected against corrosion.
- The equipotential bonding conductor should be laid in such a way as to achieve a relatively small contact area between equipotential bonding conductor and signal cables (see Figure 3-35).

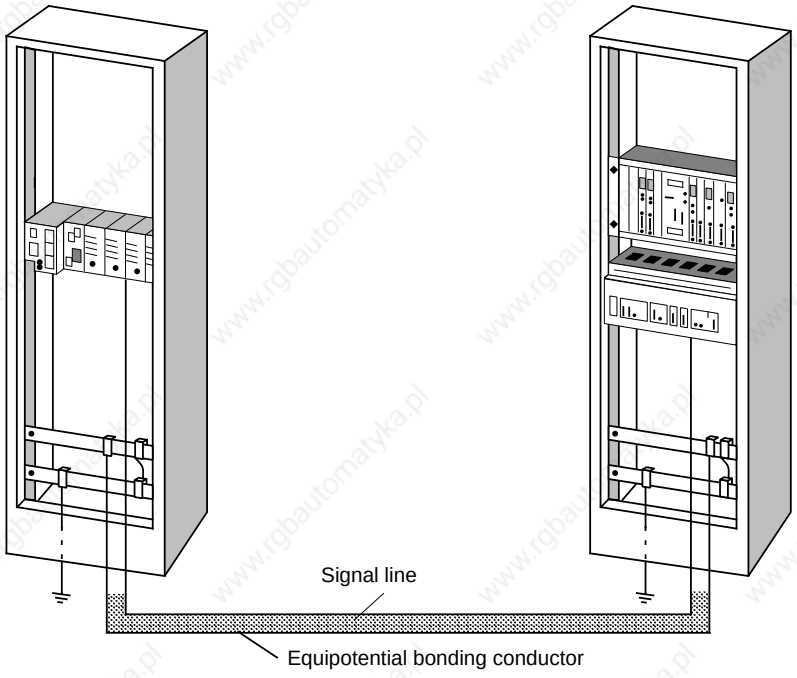



**Figure 3-35. Laying Equipotential Bonding Conductor and Signal Cable**

### 3.5.4 Shielding Cables

Shielding is a measure to weaken (attenuate) magnetic, electric or electromagnetic interference fields.

Interference currents on cable shields are discharged to ground over the shield bar which has a conductive connection to the housing. So that these interference currents do not become a source of interference in themselves, a low-resistance connection to the protective conductor is of special importance.

Use only cables with shield braiding if possible. The effectiveness of the shield should be more than 80%. Avoid cables with foil shielding since the foil can easily be damaged by tension and pressure; this leads to a reduction in the shielding effect.

As a rule, you should always shield cables at both ends. Only shielding at both ends provides good suppression in the high frequency range.

As an exception only, you can connect the shielding at one end. However, this attenuates only the lower frequencies. Shielding at one end can be of advantage in the following cases:

- If you cannot run an equipotential bonding conductor
- If you are transmitting analog signals (e.g. a few microvolts or microamps)
- If you are using foil shields (static shields).

Always use metallic or metalized connectors for data lines for serial connections. Secure the shield of the data line at the connector housing. Do ***not*** connect the shield to the PIN1 of the connector strip!
In the case of stationary operation, we recommend that you insulate the shielded cable without interruption and connect it to the shield/protective ground bar.

**Note**

If there are potential differences between the earthing points, a compensating current can flow over the shielding that is connected at both ends. In this case, connect an additional equipotential bonding conductor.

Note the following when connecting the cable shield:

- Use metal cable clamps for fixing the braided shield. The clamps have to enclose the shield over a large area and make good contact (see Figure 3-36).
- Connect the shield to a shield bar directly at the point where the cable enters the cabinet. Route the shield to the module; do **not** connect it to the module.

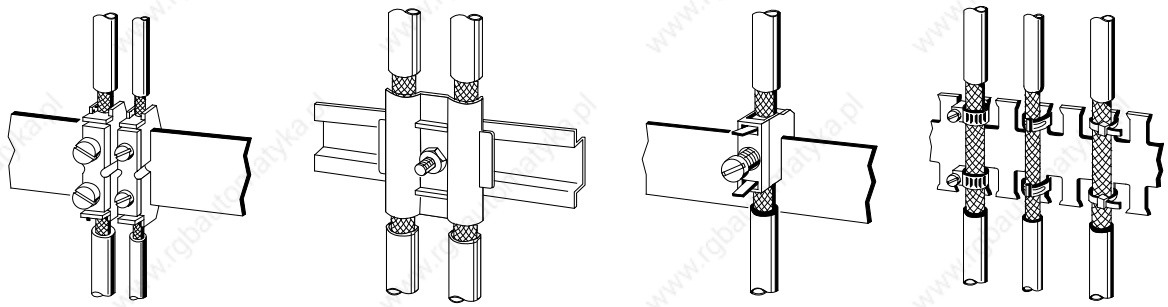


**Figure 3-36. Fixing Shielded Cables with Various Types of Cable Clamps**

## 3.5.5 Special Measures for Interference-Free Operation

**Arc Suppression Elements For Inductive Circuits**

Normally, inductive circuits (e.g. contactor or relay coils) energized by SIMATIC S5 do not need to be provided with external arc suppressing elements since the necessary suppressing elements are already integrated on the modules.

It only becomes necessary to provide arc supressing elements for inductive circuits in the following cases:

- If SIMATIC S5 output circuits can be switched off by additionally inserted contactors (e.g. relay contactors for EMERGENCY OFF). In such a case, the integral suppressing elements on the modules become ineffective.
- If the inductive circuits are ***not*** energized by SIMATIC S5.

You can use free-wheeling diodes, varistors or RC elements for wiring inductive circuits.

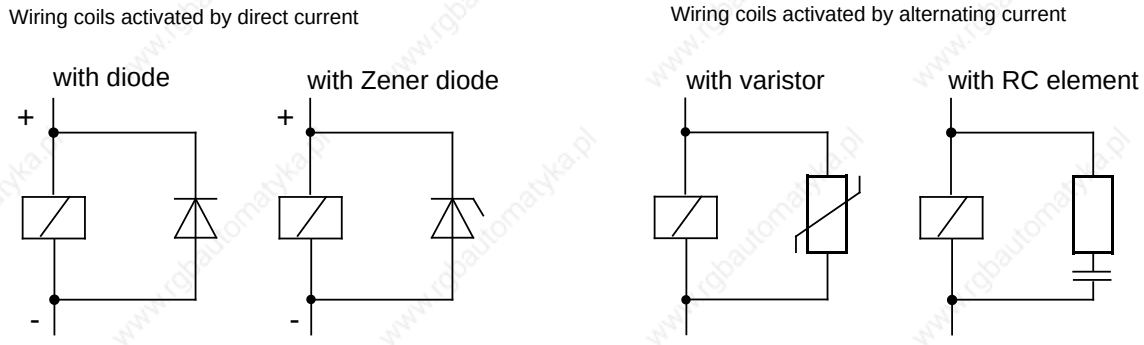



**Figure 3-37. Wiring Coils**

### Mains Connection for Programmers

Provide a power connection for the programmer in each cabinet. The plug must be supplied from the distribution line to which the protective ground for the cabinet is connected.

### Cabinet Lighting

Use, for example, LINESTRA® lamps for cabinet lighting. Avoid the use of fluorescent lamps since these generate interference fields. If you cannot do without fluorescent lamps, you must take the measures shown in Figure 3.38.

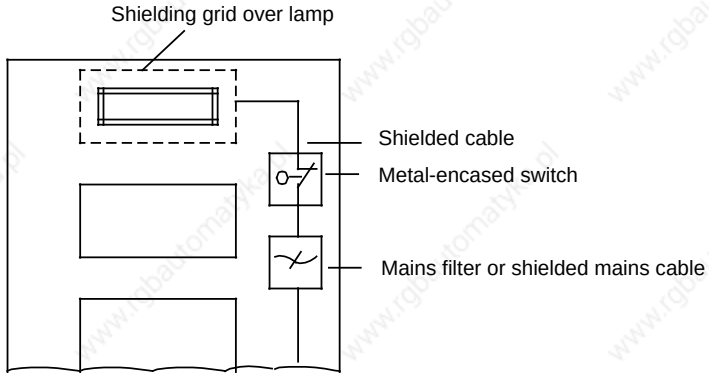



**Figure 3-38. Measures for Suppressing Interference from Fluorescent Lamps in the Cabinet**

## 4 PLC System Start-Up and Program Test


4.1 Prerequisites for Starting Up a PLC . . . . . 4 - 1

4.2 Steps for System Start-Up . . . . . 4 - 1
4.2.1 Overall Reset . . . . . 4 - 1
4.2.2 Transferring the Program . . . . . 4 - 3
4.2.3 Determining the Retentive Feature of Timers, Counters and Flags . . . . . 4 - 5

4.3 Testing the Program . . . . . 4 - 7
4.3.1 Starting the Program . . . . . 4 - 7
4.3.2 Search . . . . . 4 - 8
4.3.3 "Program Check" Test Function . . . . . 4 - 8
4.3.4 STATUS/STATUS VAR Test Function . . . . . 4 - 9
4.3.5 FORCE Outputs and Variables . . . . . 4 - 11

4.4 Special Features of the CPUs with Two Serial Interfaces . . . . . 4 - 12

4.5 Notes on the Use of Input/Output Modules . . . . . 4 - 13

4.6 System Start-Up . . . . . 4 - 14
4.6.1 Notes on Configuring and Installing a System . . . . . 4 - 14
4.6.2 System Start-Up Procedure . . . . . 4 - 15

**Figures**


4-1. Relevant Bits for Setting the Retentive Feature in System Data Word 120 . . . . . 4 - 6
4-2. Comparison of the "STATUS" and "STATUS VAR" Test Functions . . . . . 4 - 9
4-3. Representation of Signal States on a Screen (for LAD and CSF) . . . . . 4 - 10


**Tables**


4-1. Preset Retentive Feature of the CPUs 941 to 944 after Overall Reset . . . . . 4 - 5
4-2. Overview of the Functions Possible at Interface SI 1 and SI 2 . . . . . 4 - 12

# 4 PLC System Start-Up and Program Test

This chapter contains notes on starting up an S5-115U with information on testing your STEP 5 control program.
A prerequisite is knowledge of the principle of operation of the PLC (see Chapter 2).

There are notes on starting up a system at the end of the chapter.

## 4.1 Prerequisites for Starting Up a PLC

Make sure that
- all necessary I/O modules are plugged into suitable slots (see Chapter 3)
- the address assignment of the inputs and outputs is in order (see Chapter 6)
- the control program to be tested is available on the programmer.

## 4.2 Steps for System Start-Up

### 4.2.1 Overall Reset

You are recommended to perform the Overall Reset function before entering a new program. Overall Reset deletes the following:
- PLC program memory
- All data (flags, timers and counters)
- All error IDs.

In addition, all system data is automatically assigned default values after Overall Reset so that the system data area assumes a defined "basic status".

There are two ways of deleting the internal program memory:
- Offline via the switch for "Default/Overall Reset"
- Online with the "Delete" programmer function.

**Overall Reset via the Switch for "Default/Overall Reset" on the Control Panel of the CPU**

Switch on the power supply module
Set the CPU mode selector to STOP (ST)
Set the switch for "Default/Overall Reset" to the "OR" position and hold it in this position (if the switch is not held in position it will automatically spring back to the "RE" position).

While you hold down the switch for "Default/Overall Reset" in the "OR" position:
Switch the CPU mode selector twice from "ST" to "RN".

The STOP LED will momentarily go off.

Release the switch for "Default/Overall Reset".

The switch automatically springs back to the "RE" position.

The internal program memory and any RAM submodule (plugged into the CPU 941/CPU 942) are now "overall reset". After Overall Reset, the CPU automatically tests your internal program memory; in the event of a fault, the STOP LED flickers.

**Overall Reset with the "Delete" Programmer Function**

Link the programmer and the CPU over a suitable connection cable
Switch on the PLC power supply module
Set the CPU mode selector to "ST"
or set the CPU to the STOP state using the "STOP" programmer function.
Call the "Delete" auxiliary function on the programmer
Complete the input fields
in the case of "CRT-based PGs":DELETE FROM SOURCE: **PLC** BLOCK:**B** (B=all blocks)
Press the Enter key
The question "Delete?" appears on the screen
Press the Enter key

The internal program memory and any RAM submodule (plugged into the CPU 941/CPU 942) are now "overall reset". After Overall Reset, the CPU automatically tests your internal program memory; in the event of a fault, the STOP LED flickers.

### 4.2.2 Transferring the Program

There are two ways of entering the control program in the CPU:

- Transfer the program to a memory submodule
  and insert the submodule into the receptacle of the CPU. CPUs 941 and 942 process the control program direct from the submodule. CPUs 943 and 944 copy the contents of the memory submodule into the internal program memory at cold restart so that it can then be processed at extremely high speed. In this way, a PLC or a system can be started up without the need for a programmer.
- Transfer the program direct into the internal program memory of the CPU.

**Transferring the Program to a Memory Submodule**

To program a memory submodule you require a programmer with the S5-DOS "EPROM/EEPROM" package.

You must take account of the following when transferring your STEP 5 control program to a memory submodule for plugging into the submdodule receptacle of the CPU:

- Use only EPROM/EEPROM submodules; RAM submodules are only suitable for CPUs 941/942 as an expansion to the internal program memory.
- Note the following special features of the CPUs when using memory submodules 375-1LA61 and 375-1LA71:
  - EPROM 375-1LA61 (64 Kbytes) for CPU 943 and for CPU 944:
    Only 48 Kbytes of the 64-Kbyte memory capacity can be used in both CPU 943 and 944. When programming these memory submodules, note that the absolute address BFFD must not be exceeded!
  - EPROM 375-1LA71 (128 Kbytes) for CPU 944:
    An EPROM 375-1LA71, that can only be used in the CPU 944, must only be programmed to the absolute (word) address BFFD.
    The reason:
    The capacity of the internal program memory of the CPU 944 is limited to 96 Kbytes.

  Insert the memory submodule in the submodule receptacle of your programmer and program it with the S5-DOS "EPROM/EEPROM" PACKAGE. The programmer manual contains a detailed description of this package.
  After programming the memory submodule, insert it in the submodule receptacle of the CPU while the PLC is switched off.
  Switch on the power supply of the CPU.
  Overall Reset

Special features of the CPU 943/CPU 944:

After Overall Reset, the STEP 5 control program is loaded automatically from the memory submodule into the internal program memory of the CPU in the case of the CPU 943 and CPU 944.
If you do ***not*** perform Overall Reset after POWER ON, i.e. if the internal program memory still contains valid blocks, the following will happen:

- Blocks that have been loaded from E(E)PROM have the ID "Block in EPROM" in internal program memory. After POWER ON, these blocks are deleted so that they can then be loaded again.
- Blocks that do ***not*** have the ID "Block in EPROM", remain in internal program memory after POWER ON.
- Before the blocks are transferred from the memory submodule into internal program memory, the CPU deletes invalid blocks in internal program memory (Compress function)!

**Transferring the program directly to the CPU internal program memory**

If you transfer the control program directly to the CPU program memory, you must

link the programmer and CPU via a suitable connecting cable
(in the case of the CPU 943 and CPU 944 both interface SI 1 and interface SI 2 are suitable for connecting the programmer; it is essential when connecting the programmer to SI 2, that ***none*** of the following functions is activated:
ASCII driver, point-to-point master function or computer link
switch on the power supply of the PLC
test whether the backup battery has been inserted and is functional

> **Note**
>
> It is possible for an internal passivation coating to develop in new lithium batteries or in lithium batteries left unused for long periods. This coating has the effect of substantially increasing the internal resistance.
> Remedy: Depassivate the battery by loading it for approx. 2 hours with 100 ohms.

select the "on-line" mode in the default form of the "LAD, CSF, STL S5-DOS" package
select the "Transfer" auxiliary function on the programmer
specify the source (programmer or floppy disk) and destination (PLC) and initiate transfer by pressing the transfer key.

> **Note**
>
> Transfer takes place in the RUN or STOP state of the CPU. If you transfer blocks in the RUN state, you should:
> - transfer tested blocks only
> - transfer blocks in the correct order so that the CPU does not enter the STOP state (e.g. first the data blocks, then function blocks and lastly blocks which use these data and function blocks).

If blocks of the same name are already in the internal program memory of the CPU, the following message appears in the message line "... already in the PLC, overwrite?"
By pressing the transfer key again, a new block is transferred to the program memory of the CPU and the old block declared invalid. Old blocks can only be deleted using "Overall reset" or "Compress" (see Chapter 7).

**Special Points When Setting Up Data Blocks**

- Data blocks generated in the control program with the "G DB" operation are automatically dumped by the operating system direct in internal program memory.
  The contents of the data blocks can be changed using STEP 5 operations.
- Data blocks that have been transferred to an E(E)PROM submodule cannot be changed by the control program in the case of CPU 941 and CPU 942; they are suitable for e.g. fixed recipes.
  The contents of the memory submodule are copied to internal program memory after POWER ON in the case of the CPU 943 and CPU 944; this means that the contents of the data blocks can also be changed in the user program. However, after each POWER ON (and after Overall Reset), the "old" data blocks from the memory submodule are copied into internal program memory; the "current" contents are lost.
- If you want to change the contents of data blocks during execution of the control program, these data blocks must be transferred by the programmer direct over the programmer interface to the CPU in the case of the CPU 941 and CPU 942, or they are generated with the "G DB" operation.

### 4.2.3 Determining the Retentive Feature of Timers, Counters and Flags

Use the "Presetting the retentive feature/Overall Reset" switch on the operator panel of the CPU to determine the behaviour of timers, counters and flags at cold restart (both manually and automatically after power restore).
Timers, counters and flags are "retentive" if they do ***not*** lose their contents at cold restart. Those timers, counters and flags which are reset at cold restart are "nonretentive".
The following retentive feature is set as default ***after Overall Reset***:
All timers, counters and flags are nonretentive in the NR switch position. In the RE switch position, half of all timers, counters and flags are retentive:

**Table 4-1. Preset Retentive Feature of the CPUs 941 to 944 after Overall Reset**

| Switch Position | Flags | Timers | Counters |
|---|---|---|---|
| RE (retentive) | F 0.0 to F 127.7 retentive<br><br>F128.0 to 255.7 nonretentive | T0 to T63 retentive<br><br>T64 to T127 nonretentive | C0 to C63 retentive<br><br>C64 to C127 nonretentive |
| NR (nonretentive) | No retentive flags | No retentive timers | No retentive counters |

**Note**

If the battery fails on a cold restart after POWER UP and the switch is in the RE (retentive) position, the programmable controller stops (memory error).

The retentive feature in the RE switch position is determined by an entry in system data word 120 ($EAF0_H$):

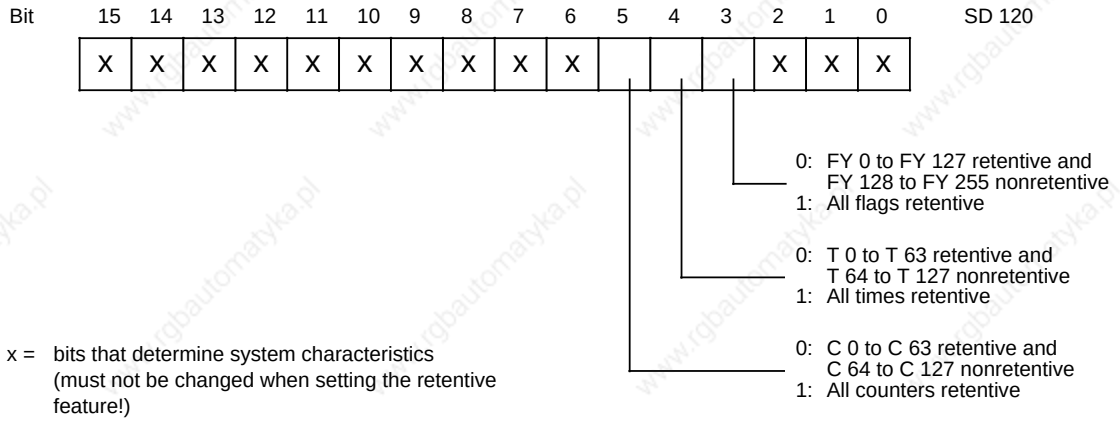



**Figure 4-1. Relevant Bits for Setting the Retentive Feature in System Data Word 120**

Bits 3, 4 and 5 of system data word 120 are set to "0" after Overall Reset of the CPU.
You can influence the retentive feature separately for flags, timers and counters by setting these bits

- in the restart program (OB20, OB21)
  or
- using the DISP ADDR programmer function (only permissible when the PLC is in the STOP state!).

You can also determine the retentive feature by setting parameters in DB1 (see Section 11.3).
In the case of Overall Reset, all timers, counters and flags are reset regardless of the switch position or the contents of system data word 120.

## 4.3 Testing the Program

The following describes the steps for starting the control program in the S5-115U. This is followed by the description of the test functions with which you can locate logical errors in program processing.

### 4.3.1 Starting the Program

Starting point: The PS 951 power supply module is switched off, the CPU mode selector is at STOP, the control program is stored on the E(E)PROM submodule.

Insert the memory submodule in the receptacle of the CPU

Switch on the power supply module

The green LEDs of the PS 951 light up (otherwise: power supply module (PS 951) defective)

(if desired:) Overall Reset of CPU

After switching on the power supply module or after Overall Reset (mode selector at STOP), the STOP LED and the BASP LED light up.
If the STOP LED ***flickers***, the CPU is defective; if the STOP LED ***flashes***, there is a fault in the memory submodule (see Chapter 5).
Online functions with the programmer over the serial interface are possible when the CPU is in the STOP state.

Set the mode selector from STOP to RUN

Both operating mode LEDs light up during the entire restart.
After the restart OB has been processed, the RUN LED lights up (cyclic program scanning).

In the event of a fault, the CPU remains in the STOP state, i.e. the STOP LED lights up. Analysis of the cause of the interrupt is described in Chapter 5.
If the control program does not work properly, you can debug the program with the "STATUS", "STATUS VAR" and "FORCE VAR" test functions.

**Note**

Test functions increase the scan time of the running control program!

### 4.3.2 Search

The "Search" function is used to find operands or symbols in the STEP 5 program.
This function makes it easier to handle longer control programs. The search function is executed differently in the individual programmers and is described in detail in the relevant manual.

### 4.3.3 "Program Check" Test Function

This programmer function causes the CPU to scan a random block step-by-step. When this function is called, program scanning is stopped at a specific point. This breakpoint (a statement in the program) is indicated by the cursor. The PLC scans the program up to the selected statement. The current signal states and the RLO are displayed up to the selected statement.
You can scan the program in sections by shifting the breakpoint as you require.

Program scanning takes place as follows:
- All jumps in the block called are traced.
- Blocks called are executed without delay.
- Program scanning is terminated automatically when block end (BE) is reached.

The following applies during Program Check:
- Neither operating mode LED is lit up.
- Inputs and outputs are not scanned. The program writes to the PIQ and reads the PII.
- All outputs are switched off. The "BASP" LED lights up.

Corrections are not possible during Program Check. However, the following test and PLC functions can be executed:
- Input and output (program modifications are possible)
- Direct signal status display (STATUS VAR)
- Forcing outputs and variables (FORCE, FORCE VAR)
- Information functions (ISTACK, BSTACK)

If the Program Check function is interrupted by PLC or program errors, the PLC goes into the STOP mode and the corresponding LED lights up on the CPU control panel.

Consult the relevant manual for information on calling the Program Check function on a programmer (not on the PG 605).

### 4.3.4 STATUS/STATUS VAR Test Function

The STATUS and STATUS VAR test functions indicate signal states of operands and the RLO. Depending on when the signal states are observed, a distinction is made between program-dependent signal status display (STATUS) and direct signal status display (STATUS VAR).

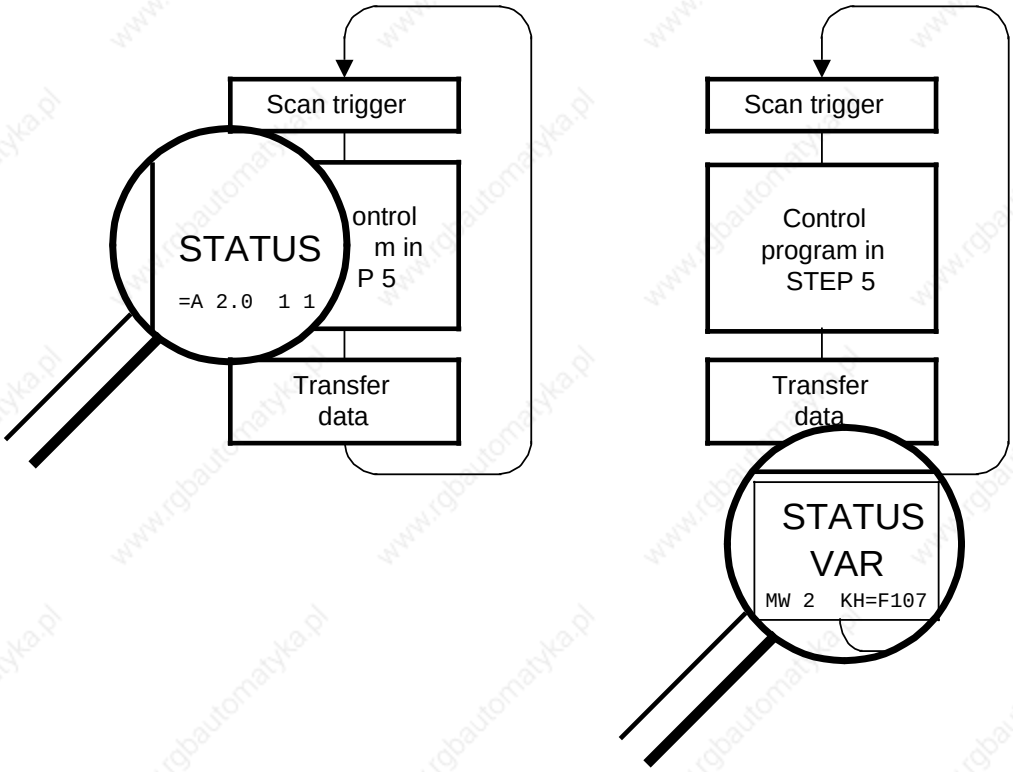



**Figure 4-2. Comparison of the "STATUS" and "STATUS VAR" Test Functions**

**Outputting Signal States on a Display Screen**

The display of signal states on a screen differs according to the following methods of representation:

**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 4-3:

| | |
|---|---|
| ═══════ | 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 4-3. Representation of Signal States on a Screen (for LAD and CSF)**

**Program-Dependent Signal Status Display "STATUS"**

Use the "STATUS" test function to indicate the current signal states and the RLO of the individual operands during program scanning.
You can also use this function to correct the program.

> **Note:**
>
> The PLC must be in the "RUN" mode for this test function.

Status processing can be interrupted by time interrupts and process interrupts. At the interrupt point, the CPU ceases to collect data for status display and transfers only data with the value 0 to the programmer instead of the data still required.
For this reason, it may occur that, when using time interrupts and process interrupts, only 0 will be flagged in the status display of a block in the following cases for a sequence of operations of varied length:

- Result of logic operation RLO
- Status/ACCUM 1
- ACCUM 2
- Status byte
- Absolute memory address SAC. "?" then appears after SAC.

Interruptions to status processing have no effect on program processing. They simply give a clear indication that the displayed data is no longer valid from the interrupt point onward.

**Direct Signal Status Display "STATUS VAR"**

Use the "STATUS VAR" test function to indicate 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. During Program Check or in the "STOP" mode, the state of the input modules is read in directly for inputs. Otherwise, only the process image of the operands that were called is displayed.

## 4.3.5 FORCE Outputs and Variables

**"FORCE" Outputs**

You can set outputs 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 either set to the Program Check function or in the "STOP" mode. The function must only be executed without the load voltage.

**"FORCE" Variables**

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 presets menu.
  - 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".

## 4.4 Special Features of the CPUs with Two Serial Interfaces

CPU 943 and CPU 944 are also available with two serial interfaces. You can connect programmers and operator panels to both interfaces. Table 4-2 shows the range of functions of the interfaces. Connection of a programmer, operator panel or SINEC L1 at SI 1 or SI 2 can cause an increase in the program execution time.

**Table 4-2. Overview of the Functions Possible at Interface SI 1 and SI 2**

| Functions at SI 1 | Functions at SI 2 |
|---|---|
| **Programmer functions**<br>no restrictions | **Programmer functions**<br>DISPL ADDR (INFO ADDR)<br>Display of memory locations; Writing back of corrected memory contents by pressing the Enter key<br>DISPLAY PLC<br>TRANSFER from PLC to FD/PG<br>Block display<br>START, STOP<br>Setting the PLC from RUN to STOP from the programmer<br>STATUS VAR, FORCE VAR<br>Test functions<br>SYSPAR<br>Display system parameters<br>INFOR DIR<br>Information about block<br>INFO DIR<br>(BLOCK: B)<br>Display block list<br>COMP<br>Compress<br>DELETE B<br>Overall Reset<br>INPUT PLC<br>TRANSFER from FD/PG to PLC<br>Entering block |
| **OP functions**<br>no restrictions | **OP functions**<br>no restrictions |
| **SINEC L1 slave**<br>no restrictions | **SINEC L1 slave**<br>no restrictions |

**Further Functions at Interface SI 2**

- Point-to-point connection (master function)
- ASCII driver
- Integral clock
- Computer link with 3964(R) procedure (only in the case of CPU 944 with the relevant operating system submodule for the purpose).

There are limitations on the simultaneous use of interfaces SI 1 and SI 2 of CPU 943 and CPU 944. Depending on the status (activity) of an interface, certain requests from a PG/OP to the other interface are not possible.

If this occurs, the function is aborted at the relevant interface by the operating system of the CPU. An error message appears informing the user that the interface function is disabled.

This message draws your attention to the fact that the other interface is currently handling a function which blocks the requested function.

Example: If "TEST STATUS" is active on SI 1, "BLOCK INPUT" is not possible at SI 2.

PG/OP functions are ***not*** possible at SI 2 if one of the following functions is active:
- ASCII driver (CPU 943/944)
- Point-to-point connection (master function in CPU 943/944)
- Computer link (3964(R) procedure); in CPU 944 with relevant operating system submodule!)

## 4.5 Notes on the Use of Input/Output Modules

**Digital Input/Output Modules**

We offer floating or nonfloating modules to suit the different signal levels. The wiring of the power supply, signal sensors and actuators is printed on the front flaps of the modules.
LEDs on the front side display the signal statuses of the inputs and outputs. The LEDs are assigned to the terminals of the front connector (see also Chapter 15, "Technical Specifications").

**Analog Input/Output Modules**

See Chapter 10 ("Analog Value Processing") for information on the use of analog modules.

> **Note**
>
> Input/output modules can only be inserted or removed when the power supply for the central controller and the signal sensors is switched off.

## 4.6 System Start-Up

The following section contains:
- Notes on configuring a system with important regulations which must be observed in order to avoid hazardous situations.
- The description of the system startup procedure.

### 4.6.1 Suggestions for Configuring and Installing the Product

A programmable controller is often used as a component in a larger system. The suggestions contained in the following warning are intended to help you safely install your programmable controller.

**Warning**

- Adhere to any safety and accident-prevention regulations applicable to your situation and system.
- If your system has a permanent power connection (stationary equipment) that is not equipped with an isolating switch and/or fuses that disconnect all poles, install either a suitable isolating switch or fuses in the building wiring system. Connect your system to a ground conductor.
- Before start-up, if you have units that operate using the main power supply, make sure that the voltage range setting on the equipment matches the local main power voltage.
- When using a 24 V supply, be sure to provide proper electric isolation between the main supply and the 24-V supply. Power supply units must be manufactured in accordance with DIN VDE 0551/EN 60742 and DIN VDE 0160.
- Fluctuations or deviations of the supply voltage from the rated value may not exceed the tolerance limit specified in the technical data. If they do, functional failures or dangerous conditions can occur in the electronic modules or equipment.
- Take suitable measures to make sure that programs that are interrupted by a voltage dip or power failure resume proper operation when the power is restored. Make sure that dangerous operating conditions do not occur even momentarily. If necessary, force an EMERGENCY OFF.
- EMERGENCY OFF devices must be in accordance with EN 60204/IEC 204 (VDE 0113) and be effective in all operating modes of the equipment. Be sure to prevent any uncontrolled or undefined restart when the EMERGENCY OFF devices are released.
- Install power supply and signal cables so that inductive and capacitive interference can not affect the automation functions.
- Install your automation system and its operative components so as to prevent unintentional operation.
- Automation equipment can assume an undefined state in the case of a wire break in the signal lines. To prevent this, take the proper hardware and software safety measures when linking the inputs and outputs of the automation equipment.

- Activation of the EMERGENCY STOP facility must create a hazard-free state for personnel and plant:
  - Actuators and drives which could cause hazardous states (e.g. main spindle drives for machine tools) must be switched off.
  - On the other hand, actuators and drives which could constitute a hazard to personnel or plant when switched off (e.g. clamping devices) must not be switched off by the EMERGENCY STOP facility.
- Activation of the EMERGENCY STOP facility must be detected by the programmable controller and evaluated in the control program.
- Connecting cables and signal cables must be installed in such a way that inductive and capacitive interference does not adversely affect the automation functions.
- Automation equipment and the operator controls for the equipment must be adequately protected against unintentional operation.
- In order that wirebreaks on the signal side cannot lead to undefined states in the automation equipment, relevant hardware and software precautions must be taken when connecting inputs and outputs.

### 4.6.2 System Start-Up Procedure

The following is a prerequisite for starting up a system:
The system and the S5-115U must not be live, i.e. the main switch must be off.

**Step 1: Visual check of the installation**; to VDE 0551, 0160 and 0113.

- Check mains voltage.
  Protective ground conductor must be connected.
- Make sure that all plugged-in modules are screwed tight to the subrack.
- Compare I/O modules plugged in with the assignment plan (note fixed or variable slot addressing).
- In the case of I/O modules, make sure that high-voltage lines (e.g. 220 V AC) do not terminate at low-voltage connectors (e.g. 24 V DC).
- When using nonfloating I/O modules, make sure that the M (0V reference) potential of the supply voltages for sensors and actuators is connected to the grounding terminal of the mounting rack ($M_{Ext}$-$M_{Int}$ connection).

**Step 2: Starting up the PLC**

- Disconnect fuses for sensors and actuators.
- Switch off the power circuits to the actuators.
- Turn on the main switch.
- Turn on the power supply.
- Switch the PLC without memory submodule to STOP.
- Connect the programmer to the CPU.

After the power switch is turned on, the green LEDs light up on the power supply and the red ST (STOP) LED lights up on the CPU.

- OVERALL RESET of the PLC.
- Transfer the program in the case of RAM operation.
- Switch the PLC to "RUN".

The red ST LED goes out and the green "RN" LED lights up.

**Step 3: Testing the signal inputs (peripheral)**

- Insert the fuse for the signal sensors. Leave the fuses for the actuators and the power circuits disconnected.
- Activate all sensors in sequence.
- You can scan all inputs using the "STATUS VAR" programmer function.

If the sensors function properly and their signals are received, the appropriate LEDs must light up on the I/O module.

**Step 4: Testing the signal outputs (peripheral)**

- Insert the fuse for the actuators. Leave the power circuits of the actuators disconnected.
- You can force each output using the "FORCE VAR" programmer function.

The LEDs of the forced outputs must light up and the circuit states of the corresponding actuators must change.

**Step 5: Entering, testing and starting the program**

Leave the power circuits for the actuators disconnected.

- Enter the program using the "INPUT" programmer function. You can enter the program in the "STOP" or "RUN" mode.

The red "ST" LED or the green "RN" LED lights up. A battery must be installed if a RAM submodule is used.

- Test the program block by block and make any necessary corrections.
- Dump the program in a memory submodule (if desired).
- Switch the PLC to "STOP"
- Switch on the power circuits for the actuators.
- Switch the PLC to "RUN".

The green "RN" LED lights up and the PLC scans the program.

## 5 Error Diagnostics


5.1 Interrupt Analysis . . . . . 5 .- 2
5.1.1 "ISTACK" Analysis . . . . . 5 .- 2
5.1.2 Meaning of the ISTACK Displays . . . . . 5 - 6
5.1.3 LED Error Signalling . . . . . 5 .- 9
5.1.4 Error Messages When Using Memory Submodules (only in the case of CPU 943/944) . . . . . 5 -10

5.2 Program Errors . . . . . 5 .- 11
5.2.1 Determining an Error Address . . . . . 5 - 12
5.2.2 Program Trace with the Block Stack ("BSTACK") Function (not possible on the PG 605U programmer) . . . . . 5 - 15

5.3 Other Causes of Malfunction . . . . . 5 - 16

5.4 System Parameters . . . . . 5 .- 16

## Figures


5-1. Structured Program with Illegal Statement . . . . . 5 - 12
5-2. Addresses in the CPU Program Memory . . . . . 5 - 13
5-3. Calculating an Error's Relative Address . . . . . 5 - 14
5-4. Program Trace with the "BSTACK" Function . . . . . 5 - 15


## Tables


5-1. General Error Analysis . . . . . 5 - 1
5-2. ISTACK Display on PG 605U . . . . . 5 - 3
5-3. Control Bit Display . . . . . 5 - 5
5-4. Interrupt Stack Display . . . . . 5 - 5
5-5. Meaning of the ISTACK Displays . . . . . 5 - 6
5-6. Mnemonics for Control Bits and Interrupt Display . . . . . 5 - 8
5-7. Meaning of the Error LEDs on the CPUs . . . . . 5 - 9
5-8. Error when Using Memory Submodules (CPU 943 and CPU 944) . . . . . 5 - 10
5-9. Program Errors . . . . . 5 - 11
5-10. Other Causes of Malfunction . . . . . 5 - 16

# 5 Error Diagnostics

Malfunctions in the S5-115U can have various causes. If the PLC malfunctions, first determine whether the problem is in the CPU, the program, or the I/O modules (see Table 5-1).

**Table 5-1. General Error Analysis**

| Fault/Error Condition | Fault/Error Analysis |
|---|---|
| The CPU is in the "STOP" mode.<br>The red LED is lit up. | The problem is in the CPU.<br>Perform an interrupt analysis with the programmer<br>(see Section 5-1). |
| The CPU is in the "RUN" mode.<br>The green LED is lit up.<br>Operation is faulty. | There is a program error. Determine the error address<br>(see Section 5-2).<br>There is an I/O problem. Perform a malfunction analysis<br>(see Section 5-3). |

**Note**

To make a general distinction between PLC and program errors, program OB1 with "BE" as the first statement. A properly functioning PLC enters the "RUN" mode on a Cold Restart.

**Caution**

There are risks involved in changing the internal program memory direct with the DISPLAY DIR programmer function.
For example, if the CPU is in RUN mode, memory areas (e.g. BSTACK) may be over-written causing the CPU to "crash".
Take the following measures to avoid such risks:

- Change only the system data area documented in this manual
- Use only the control program to change the system data area!

## 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 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 can read only part of the ISTACK when the PLC is in the "RUN" mode.

The following tables show which control bits and which malfunction causes are reported in the ISTACK. The system data words containing the ISTACK bits are also specified.
See the subsequent tables for an explanation of the abbreviations or error codes used here.

**ISTACK display on the PG 605U**

The following table shows which bits in the ISTACK are relevant for error diagnostics. The bits in boxes with heavy borders indicate the cause of a malfunction and the step address counter.

**Table 5-2. ISTACK Display on PG 605U**

| Bit / Byte | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Absolute addr. | System data word (RS) |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | | | BST SCH | SCH TAE | ADR BAU | SPA BBR | | | EA0A | SD 5 |
| 2 | CA-DA | CE-DA | | REM AN | | | | | EA0B | |
| 3 | STO ZUS | STO ANZ | NEU STA | | BAT PUF | | BARB | BARB END | EA0C | SD 6 |
| 4 | | UA FEHL | | | | AF | | | EA0D | |
| 5 | ASP NEP | ASP NRA | KOPF NI | | ASP NEEP | | | | EA0E | SD 7 |
| 6 | KEIN AS | SYN FEH | NINEU | | | | | UR LAD | EA0F | |
| 7 | IRRELEVANT | | | | | | | | | |
| 8 | IRRELEVANT | | | | | | | | | |
| 9 | STOPS | | SUF | TRAF | NNN | STS | STUEB | FEST | EBAC | SD 214 (UAW) |
| 10 | NAU | QVZ | KOLIF | ZYK | SYSFE | PEU | BAU | ASPFA | EBAD | |
| 11 | | | | | | | | | EBAA | SD 213 |
| 12 | ANZ1 | ANZ0 | OVFL | | OR | STA TUS | VKE | ERAB | EBAB | |
| 13 | 6th nesting level | | | | | OR | VKE | FKT | EBA8 | SD 212 |
| 14 | IRRELEVANT | | | | | | | | EBA9 | |
| 15 | 4th nesting level | | | | | OR | VKE | FKT | EBA6 | SD 211 |
| 16 | 5th nesting level | | | | | OR | VKE | FKT | EBA7 | |

**Table 5-2. ISTACK Display on PG 605U (Continued)**

| Byte \ Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Absolute addr. | System data word (SD) |
|---|---|---|---|---|---|---|---|---|---|---|
| 17 | 2nd nesting level | | | | | OR | VKE | FKT | EBA4 | SD 210 |
| 18 | 3rd nesting level | | | | | OR | VKE | FKT | EBA5 | |
| 19 | Nesting depth (0 to 6) | | | | | | | | EBA2 | SD 209 |
| 20 | 1st nesting level | | | | | OR | VKE | FKT | EBA3 | |
| 21 | Start address of the data block (high) | | | | | | | | EBA0 | SD 208 |
| 22 | Start address of the data block (low) | | | | | | | | EBA1 | |
| 23 | Block stack pointer (high) | | | | | | | | EB9E | SD 207 |
| 24 | Block stack pointer (low) | | | | | | | | EB9F | |
| 25 | Step address counter (high) | | | | | | | | EB9C | SD 206 |
| 26 | Step address counter (low) [1] | | | | | | | | EB9D | |
| 27 | Statement register (high) | | | | | | | | EB9A | SD 205 |
| 28 | Statement register (low) | | | | | | | | EB9B | |
| 29 | ACCUM 2 (high) | | | | | | | | EB98 | SD 204 |
| 30 | ACCUM 2 (low) | | | | | | | | EB99 | |
| 31 | ACCUM 1 (high) | | | | | | | | EB96 | SD 203 |
| 32 | ACCUM 1 (low) | | | | | | | | EB97 | |

1 Absolute memory address of the next statement that still has not been scanned.

### ISTACK Display on the PG 635/670/675/685/695 and 750

Tables 5-3 and 5-4 show the ISTACK as it is displayed on CRT-based programmers. Relevant information for the S5-115U is in bold print.

**Table 5-3. Control Bit Display**

| **CONTROL BITS** | | | | | | | | Absolute address | System data word |
|---|---|---|---|---|---|---|---|---|---|
| NB | PBSSCH | **BSTSCH** | **SCHTAE** | **ADRBAU** | **SPABBR** | NAUAS | QUITT | EA0A | SD5 |
| **CA-DA** | **CE-DE** | NB | **REMAN** | NB | NB | NB | NB | EA0B | |
| **STOZUS** | **STOANZ** | **NEUSTA** | NB | **BATPUF** | NB | **BARB** | **BARBEND** | EA0C | SD6 |
| NB | UAFEHL | MAFEHL | EOVH | NB | **AF** | NB | NB | EA0D | |
| **ASPNEP** | **ASPNRA** | **KOPFNI** | PROEND | **ASPNEEP** | PADRFE | ASPLUE | RAMADFE | EA0E | SD7 |
| **KEINAS** | **SYNFEH** | **NINEU** | NB | NB | NB | SUMF | **URLAD** | EA0F | |

**Table 5-4. Interrupt Stack Display**

| **INTERRUPT STACK** | | | | | | | | | | | | | | Abso-lute addr. | System data word |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DEPTH: | 01 | | | | | | | | | | | | | | |
| **BEF-REG:**<br>**BST-STP:** | 0000<br>EB07 | SAZ:<br>Baust.-NR.:<br>REL-SAZ: | E30A<br>1<br>0000 | DB-ADR:<br>DB-NR.: | 0000 | | | | | | | | | EB9A<br>-EBA0 | SD205-208 |
| **AKKU1:** | FFF1 | **AKKU2:** | 00FF | | | | | | | | | | | EB96-<br>EB98 | SD203-204 |
| BRACKETS: | **KE1** | 000 | **KE2** | 000 | **KE3** | 000 | **KE4** | 000 | **KE5** | 000 | **KE6** | 000 | | EBA2-<br>EBA8 | SD209-212 |
| RESULT DISPLAY: | **ANZ1** | **ANZ0** | **OVFL** | CARRY | **ODER** | **STATUS** | **VKE** | **ERAB** | | | | | | EBAA | SD213 |
| CAUSE OF FAULT: | **STOPS** | | **SUF** | **TRAF** | **NNN** | **STS** | **STUEB** | | | | | | | EBAC | SD214 (UAW) |
| | **NAU** | **QVZ** | | **ZYK** | | **PEU** | **BAU** | **ASPFA** | | | | | | EBAD | |

## 5.1.2 Meaning of the ISTACK Displays

Use Table 5-5 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 5-5. Meaning of the ISTACK Displays**

| Fault/Error | Fault/Error Cause (ID in ISTACK) | Cause | Remedy |
|---|---|---|---|
| Cold Restart is not possible. | NINEU SYNFEH/ KOPFNI | Faulty block:<br>• Compressing has been interrupted by a power failure.<br>• Block transfer between programmer and PLC was interrupted by a power failure.<br>• Program error (TIR/TNB/BMW) | Perform an Overall Reset.<br>Reload the program. |
| | KOLIF | DB1 is programmed incorrectly. | Check the following:<br>• ID for interprocessor communication flag definitions ("MASK01");(see Section 12.1)<br>• ID for the part of DB1 to be interpreted ("DB1");(see Section 11.3)<br>• the end IDs in each case for interprocessor communication flag definitions or for the part of DB1 to be interpreted |
| | FEST | There is an error in the self-test routine of the CPU. | Replace the CPU. |
| Faulty submodule | ASPFA | The submodule ID is illegal:<br>(AG 110S/135U/150U submodule) | Plug in the correct submodule. |
| Battery failure | BAU | There is no battery or the battery is low and the retentive feature is required. | Replace the battery.<br>Perform an Overall Reset.<br>Reload the program. |
| I/Os not ready | PEU | The I/Os are not ready:<br>• There has been a power failure in the expansion unit.<br>• The connection to the expansion unit has been interrupted.<br>• There is no terminator in the central controller. | • Check the power supply in the expansion unit.<br>• Check the connection.<br>• Install a terminator in the central controller. |
| Program scanning interrupted | STOPS | The mode selector is on STOP. | Put the mode selector on RUN. |

**Table 5-5. Meaning of ISTACK Displays (Continued)**

| Error/Fault | Fault/Error Cause (ID in ISTACK) | Cause | Remedy |
|---|---|---|---|
| Program scanning interrupted | SUF | Substitution error:<br>A function block was called with an incorrect actual parameter. | Correct the function block call. |
| | TRAF | Transfer error:<br>- A data block statement has been programmed with data word number greater than the data block length.<br>- A data block statement has been programmed without opening a DB first.<br>- DB to be generated is too long for user memory (G DB operation) | Correct the program error. |
| | STS | - Software stop by statement (STP, STS)<br>- STOP request from programmer<br>- STOP request from SINEC L1 master | |
| | NNN | - A statement cannot be decoded.<br>- A parameter has been exceeded. | Correct the program error. |
| | STUEB | Block stack overflow:<br>- The maximum block call nesting depth of 32 has been exceeded.<br>- Interupt-driven or time-driven program interrupts cyclic program during processing of an integrated function block and an integrated function block is also called in the interrupting program. | Correct the program error.<br><br>Disable interrupts in the cyclic program before calling integrated function blocks. |
| | NAU | There has been a power failure. | |
| | QVZ | Time-out from I/Os:<br>- A peripheral byte that was not addressed has been referenced in the program.<br>- An I/O module does not acknowledge. | Correct the program error or replace the I/O module. |
| | ZYK | Scan time exceeded:<br>The program scanning time is greater than the set monitoring time. | Check the program for continuous loops. If necessary, retrigger the scan time with OB31 or change the monitoring time. |

**Table 5-6. Mnemonics for Control Bits and Interrupt Display**

| Control Bit Mnemonics | | Mnemonics for Cause of Error/Fault (=Error ID) | |
|---|---|---|---|
| BSTSCH | Block shift requested | STOPS | Interrupt display word |
| SCHTAE | Block shift active (function: KOMP:AG) | SUF | Substitution error |
| | | 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 S5-115U (e.g., a 150S statement). |
| SPABBR | Compress operation aborted | STS | Operation interrupted by a programmer STOP request or programmed STOP statements. |
| CA-DA | Interprocessor communication flag output address list available | STUEB | Block stack overflow: The maximum block call nesting depth of 16 (or 32 in the case of CPU 944) has been exceeded. |
| CE-DE | Interprocessor communication flag input address list available | FEST | Error in the CPU self-test routine |
| REMAN | 0: all timers, counters, and flags are reset on Cold Restart | NAU | Power failure |
| | | QVZ | Time-out from I/Os: A nonexistent module has been referenced. |
| | 1: the second half of timers, counters, and flags are reset on Cold Restart | KOLIF | Interprocessor communication flag transfer list is incorrect. |
| STOZUS | "STOP" state (external request for example via the programmer) | ZYK | Scan time exceeded: The set maximum permissible program scan time has been exceeded. |
| STOANZ | "STOP" display | SYSFE | Error in DB1 |
| NEUSTA | PC in Cold Restart | PEU | I/Os not ready: power failure in the I/O expansion unit; connection to the I/O expansion unit interrupted |
| BATPUF | Battery backup okay | | No terminator in the central controller |
| BARB | Program check | BAU | Battery failure |
| BARBEND | Request for end of program check | ASPFA | Illegal memory submodule |
| UAFEHL | Incorrect interrupt display | | |
| AF | Interrupt enable | | |
| ASPNEP | Memory submodule is an EPROM | | |
| ASPNRA | Memory submodule is a RAM | | |
| KOPFNI | Block header cannot be interpreted | | |
| ASPNEEP | Memory submodule is an EEPROM | | |
| KEINAS | No memory submodule | | |
| SYNFEH | Synchronization error (blocks are incorrect) | Other mnemonics: | |
| | | UAW | Interrupt display word |
| NINEU | Cold Restart not possible | ANZ1/ANZ0 | 00: ACCUM 1=0 or 0 is shifted |
| URLAD | Bootstrapping required | | 01: ACCUM 1>0 or 1 is shifted |
| | | | 10: ACCUM 1<0 |
| | | OVFL | Arithmetic overflow (+or -) |
| | | ODER (OR) | OR memory (set by "O" operation) |
| | | STATUS | STATUS of the operand of the last binary statement executed |
| | | VKE | Result of logic operation |
| Other mnemonics: | | ERAB | First scan |
| SD | System data (from address $EA00_H$) | KE1...KE6 | Nesting stack entry 1 to 6 entered for A( and O( |
| | | FKT | 0 : O( |
| | | | 1 : A( |
| | | BEF-REG | Statement register |
| | | SAZ | Step address counter |
| | | DB-ADR | Data block address |
| | | BST-STP | Block stack pointer |
| | | NR | Block number (OB, PB, FB, SB, DB) |
| | | REL-SAZ | Relative step address counter |

### 5.1.3 LED Error Signalling

Certain errors are indicated by LEDs on the CPU depending on its design. Table 5-7 explains these error signals.

**Table 5-7. Meaning of the Error LEDs on the CPUs**

| LED | Meaning |
|---|---|
| QVZ lights up | Timeout (CPU went into STOP mode) |
| ZYK lights up | Scan time exceeded (CPU went into STOP mode) |
| BASP lights up | Digital outputs are disabled (CPU is in RESTART or STOP mode) |
| Red STOP LED is flashing | Memory error (block structure damaged)<br><br>After CPU COLD RESTART and after POWER UP, a memory error message may result if the user program contains the TNB, TIR or TDI operations. It is possible to inadvertently overwrite the following with these operations:<br>• Block headers<br>• Memory areas designated as "free" by the operating system.<br>The operating system writes the erroneous address found when generating the address list into system data word 103 ($EACE_H$). You can display the contents of the system data locations using the DISP ADDR programmer function.<br><br>**Special features of CPU 943 and CPU 944:**<br>Error in the use of memory submodules (see Section 5.8)<br><br>**Special features of CPU 944:**<br>You can detect whether the error in the address list has occurred in memory bank 1 or memory bank 2 by using the location counters in the two memory banks.<br>• Location counter for memory bank 1: system data word 33 ($EA42_H$)<br>• Location counter for memory bank 2: system data word 32 ($EA40_H$).<br><br>Since the location counter is updated only after the address list has been generated, the location counter of the memory bank with the erroneous address points to its initial value;<br>$1000_H$ in the case of memory bank 1,<br>$1001_H$ in the case of memory bank 2.<br>If system data word 33 contains the value "$1000_H$", after a memory error message, the address in system data word 103 refers to memory bank 1.<br>The value of the erroneous address can be displayed and interpreted using the DISPL. ADDR. programmer function. The programmer automatically accesses memory bank 1. |
| Red STOP LED is flickering | No error on first plugging in CPU<br>Remedy: Overall Reset<br>or<br>Error in the CPU self-test routine<br>Remedy: Exchange CPU |

### 5.1.4 Error Messages When Using Memory Submodules (only in the case of CPU 943/944)

A flashing red LED (STOP LED) indicates errors when memory submodule blocks are loaded into the internal RAM. The cause of error is stored in system data word 102.

**Table 5-8. Error when Using Memory Submodules (CPU 943 and CPU 944)**

| Cause of Error | Display in SD 102 (EACC) | Display in ISTACK | Remedy |
|---|---|---|---|
| Invalid memory submodule. | $F002_H$ | URLAD ASPFA | Use the appropriate submodule. |
| Faulty contents of memory submodule. | $F003_H$ | URLAD | Delete and reprogram the submodule. |
| More than 48 Kbytes have been programmed for the 375-1LA**61** memory submodule.<br>Only in the case of CPU 944: More than 96 Kbytes have been programmed for the 375-1LA**71** memory submodule. | $F001_H$ | URLAD | Shorten program. |
| All blocks cannot be copied. | $F001_H$ | URLAD | The internal memory already contains blocks. Check to see if these blocks are needed. Delete the blocks or optimize the program. |
| Only in the case of CPU 944: More than 48 Kbytes have been programmed in data blocks intended for memory bank 1. | $F004_H$ | URLAD | Shorten the data blocks. |
| User program operations TNB, TIR or TDI have overwritten block headers or free memory space. | $0000_H$ | NINEU<br>SYNFEH | See Table 5-7. |

**Note**

A maximum of 48 Kbytes can be used in the case of the memory submodule 6ES5 375-1LA61.

## 5.2 Program Errors

Table 5-9 lists malfunctions caused by program errors.

**Table 5-9. Program Errors**

| Error | Action |
|---|---|
| All inputs are zero | Check the program |
| All outputs are not set | |
| One input is zero. One output is not set | Check program assignments (double assignment, edge formation) |
| Timer or counter is not running or is incorrect | |
| Cold Restart is faulty | Check Cold Restart blocks OB21/OB22 or insert them |
| Sporadic malfunctions occur | Check the program with STATUS |

### 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 went into the "STOP" mode.

You can use the "DIR PC" programmer function to determine the appropriate block start address.

**Example**:

Figure 5-1 shows a program consisting of OB1, PB0, and PB7. PB7 contains an illegal statement.

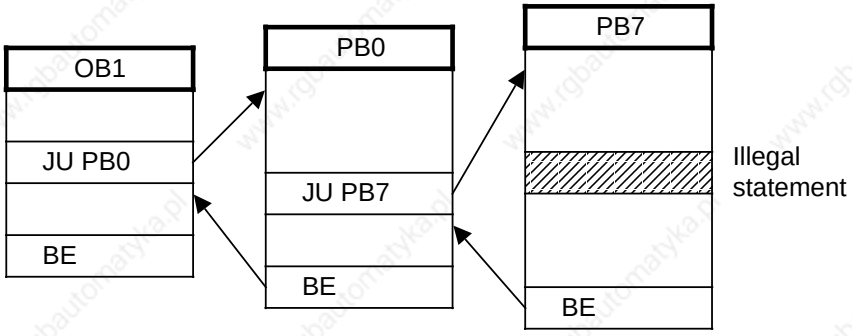



**Figure 5-1. Structured Program with Illegal Statement**

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 (see Figure 5-2).

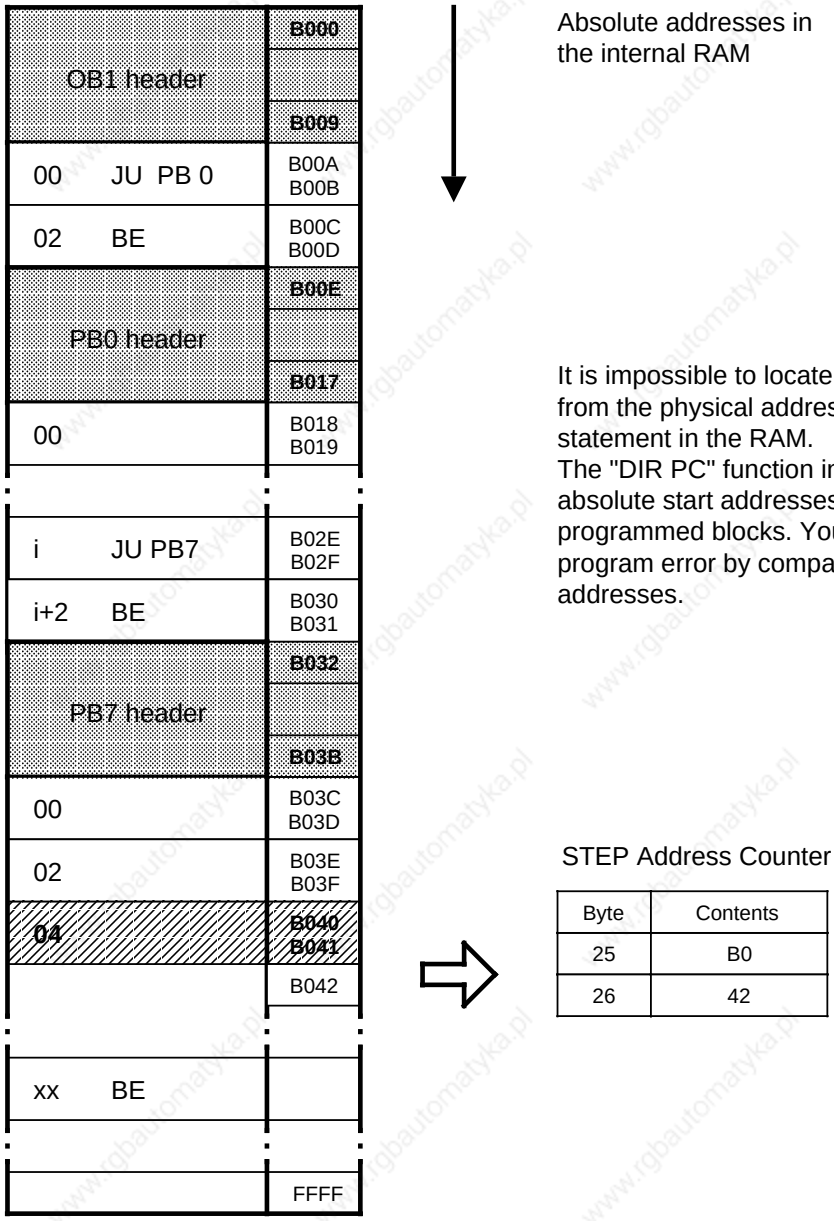



**Figure 5-2. Addresses in the CPU Program Memory**

**Address Calculation (necessary only when using the PG 605U programmer)**

To make program corrections, you need the address of the statement that caused the malfunction, relative to the particular block (relative address).
Determine the faulty block by comparing the STEP address counter value (SAZ value) and the "DIR PC" display.
The difference between the SAZ value and the block start address gives the relative address of the error (see Figure 5-3).

| **ISTACK Byte** | 25 | 26 |
|---|---|---|
| **STEP Address Counter** | B0 | 42 |

The absolute address B042 is greater than the start address of PB7.
Therefore, the illegal statement is in PB7.

| **DIR PC** | |
|---|---|
| Block | Start Address |
| PB0 | B018 |
| PB7 | B03C |
| OB1 | B00A |

**Calculation of the relative address:** B042 - B03C=0006

Consequently, "0006" is the address of the statement in PB7 **before** which the CPU went into the "STOP" mode.

**Figure 5-3. Calculating an Error's Relative Address**

## Note

The programmers (with the exception of the PG 605) calculate the relative error address automatically and display it in the ISTACK.
The interruption point, block type, block number and relative address calculated by the programmers to be the cause of error may, under certain circumstances, be wrongly calculated and displayed in the case of the CPU 944*****.
For this reason, the CPU 944 writes the program interruption point into DB0, parallel to the ISTACK entry. DB0 is generated automatically by the CPU 944. You can detect the correct interruption point with the help of the "STATUS VAR" and "FORCE VAR" programmer functions.

Example: STATUS VAR/FORCE VAR screen form on the programmer

```
DB 0
DW 0   KS =  ..     Block type in KS format
DW 1   KF =  ...    Block number in KF format
DW 2   KH =  ....   Relative address of the interruption point in the
                    block
```

* Error removed in the case of:
- S5-DOS Stage 3, basic package V1.1

**Display of an Illegal Statement**

You can use the "SEARCH RUN" programmer function to find specific program locations (see Section 8.3). You can use this function to look for the relative address of an error.

### 5.2.2 Program Trace with the Block Stack ("BSTACK") Function (not possible on the PG 605U programmer)

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 FB2. The CPU went into the "STOP" mode with the error message "TRAF" (because of incorrect access. DB5 is two words long. DB3 is ten words long).
You can use the "BSTACK" function to determine the path used to reach FB2 and to determine which block has passed the wrong parameter. The "BSTACK" contains the three return addresses (as marked in Figure 5-4).

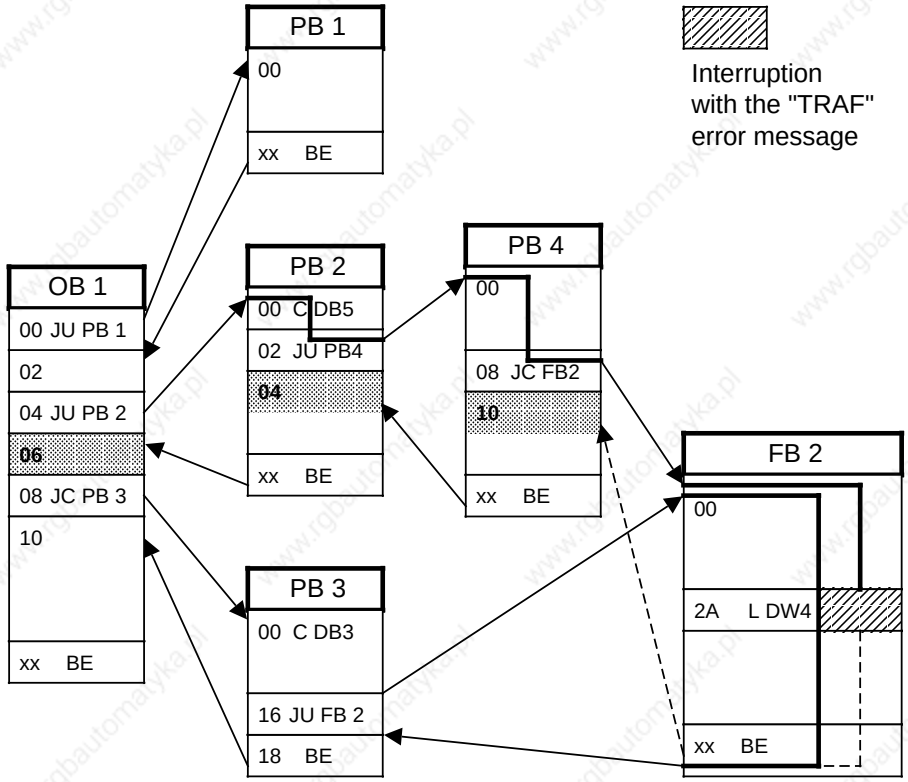



**Figure 5-4. Program Trace with the "BSTACK" Function**

## 5.3 Other Causes of Malfunction

Hardware components or improper installation can also cause malfunctions. Table 5-10 summarizes such malfunctions.

**Table 5-10. Other Causes of Malfunction**

| Fault/Error | Action |
|---|---|
| All inputs are zero. | Check the module and the load voltage. |
| All outputs are not set. | |
| One input is zero. One output is not set. | |
| The green LEDs on the power supply module do not light up. | Check the module and replace it if necessary. |
| Sporadic malfunctions occur. | Check the memory submodule. Check to see if the controller has been set up according to EMC guidelines. |
| The PLC will not go into the "RUN" mode. | Perform an Overall Reset. |

**Note**

If the PLC still does not operate properly after you have taken the appropriate action recommended in Table 5-10, try to determine the faulty component by replacement.

## 5.4 System Parameters

Use the "SYSPAR" programmer function to read the system parameters (e.g. software release) out of the CPU.

## 6 Addressing/Address Assignments


6.1 Address Structure . . . . . 6 - 1
6.1.1 Digital Module Addresses . . . . . 6 - 1
6.1.2 Analog Module Addresses . . . . . 6 - 1

6.2 Slot Address Assignments . . . . . 6 - 1
6.2.1 Fixed Slot Address Assignments . . . . . 6 - 2
6.2.2 Variable Slot Address Assignments . . . . . 6 - 3

6.3 Handling Process Signals . . . . . 6 - 7
6.3.1 Accessing the PII . . . . . 6 - 8
6.3.2 Accessing the PIQ . . . . . 6 - 9
6.3.3 Direct Access . . . . . 6 - 10

6.4 Address Allocation on the Central Processing Units . . . . . 6 - 11

## Figures


6-1. Format of a Digital Address . . . . . 6 - 1
6-2. Fixed Slot Addressing in the Central Controllers . . . . . 6 - 2
6-3. Fixed Slot Addressing in the Expansion Unit . . . . . 6 - 3
6-4. Setting Addresses on the Addressing Panel of the IM 306 Interface Module . . . . . 6 - 4
6-5. Setting a DIP Switch . . . . . 6 - 5
6-6. Addresses of the Input/Output Modules . . . . . 6 - 7
6-7. Location of the Process Images . . . . . 6 - 7
6-8. Accessing the PII . . . . . 6 - 8
6-9. Accessing the PIQ . . . . . 6 - 9
6-10. Loading Input/Output Modules . . . . . 6 - 10
6-11. Memory Allocation on the CPU . . . . . 6 - 12
6-12. Address Assignment in the I/O Area . . . . . 6 - 15


## Tables


6-1. Address Allocation in the System Data Area . . . . . 6 - 16
6-2. Address Allocation in the Flag, Timer and Counter Areas . . . . . 6 - 18
6-3. Block Address List . . . . . 6 - 18

# 6 Addressing/Address Assignments

In order to be able to access input/output modules, these modules must be assigned addresses.

## 6.1 Address Structure

Digital modules generally are addressed by bit, analog modules by byte or word. Consequently, their addresses have different formats.

### 6.1.1 Digital Module Addresses

One bit represents a channel on a digital module. You must therefore assign a number to each bit. When numbering, note the following:

- The CPU program memory is divided into different address areas (see Section 6.3).
- Number individual bytes consecutively in relation to the start address of the relevant address area.
- Number the eight bits of each byte consecutively (0 to 7).

Figure 6-1 shows the format of a digital address:

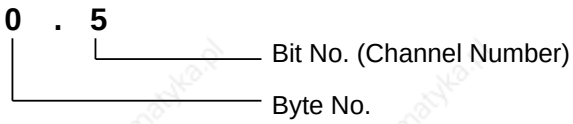



**Figure 6-1. Format of a Digital Address**

### 6.1.2 Analog Module Addresses

Each channel of an analog module is represented by two bytes (=one word).
An analog channel address is thus represented by the number of the high-order byte.

## 6.2 Slot Address Assignments

You can establish addresses for S5-115U modules in the following two ways:

- Fixed Slot Addressing
  Each slot has a fixed address under which you can reference the module plugged into that slot.
- Variable Slot Addressing
  The user can specify an address for each slot.

Fixed slot and variable slot addresses are relevant only for modules of block design. The addresses of intelligent I/O modules and modules of ES 902 design (S5-135U/155U) are set on the modules themselves. In this case, the address need not be set on the IM 306.

### 6.2.1 Fixed Slot Address Assignments

Input/output modules are referenced under permanently assigned slot addresses when the following conditions exist for the S5-115U:

- The PLC is operated without an expansion unit interface module and a terminating resistor is used.
- The PLC is operated with the IM 305 interface module (centralized configuration, see Section 3.2.5).

The number of address bytes available for digital and analog modules varies.

**Digital Modules**

Each slot has four bytes, so that 32 binary inputs or outputs can be addressed. If you plug in digital modules with 8 or 16 channels, use the low-order byte numbers for addressing. In this case, the high-order byte numbers are irrelevant.

**Analog Modules**

For fixed slot addressing, analog modules can be plugged into slots 0 to 3 of a central controller only.
Each slot has 32 bytes. You can thus address 16 analog channels. If you plug in 8-channel modules, use the 16 low-order byte numbers for addressing. In this case, the 16 high-order byte numbers are irrelevant.

**Note the following:**

- Input and output modules cannot have the same address.
- If an analog module has been assigned an address for a particular slot, this address cannot be used for digital modules and vice versa.

Figures 6-2 and 6-3 show the exact assignment of fixed addresses (please observe the "Installation Guidelines" in Sections 3.1.1 and 3.1.2).

| Slot numbers in the central controller | PS | CPU | 0 | 1 | 2 | 3 | 4 | 5 | 6 | IM |
|---|---|---|---|---|---|---|---|---|---|---|
| Digital modules | | | 0.0 . . . 3.7 | 4.0 . . . 7.7 | 8.0 . . . 11.7 | 12.0 . . . 15.7 | 16.0 . . . 19.7 | 20.0 . . . 23.7 | 24.0 . . . 27.7 | |
| Analog modules | | | 128 . . . 159 | 160 . . . 191 | 192 . . . 223 | 224 . . . 255 | Analog modules cannot be plugged into these slots | | | |
| **Modules** | **Addresses** | | | | | | | | | |

**Figure 6-2. Fixed Slot Addressing in the Central Controllers**

| Slot numbers in the expansion unit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | IM |
|---|---|---|---|---|---|---|---|---|---|---|
| Digital modules | 28.0 . . 31.7 | 32.0 . . 35.7 | 36.0 . . 39.7 | 40.0 . . 43.7 | 44.0 . . 47.7 | 48.0 . . 51.7 | 52.0 . . 55.7 | 56.0 . . 59.7 | 60.0 . . 63.7 | IM 305 |
| Analog modules | Analog modules cannot be plugged in here. | | | | | | | | | |
| **Modules** | **Addresses** | | | | | | | | | |

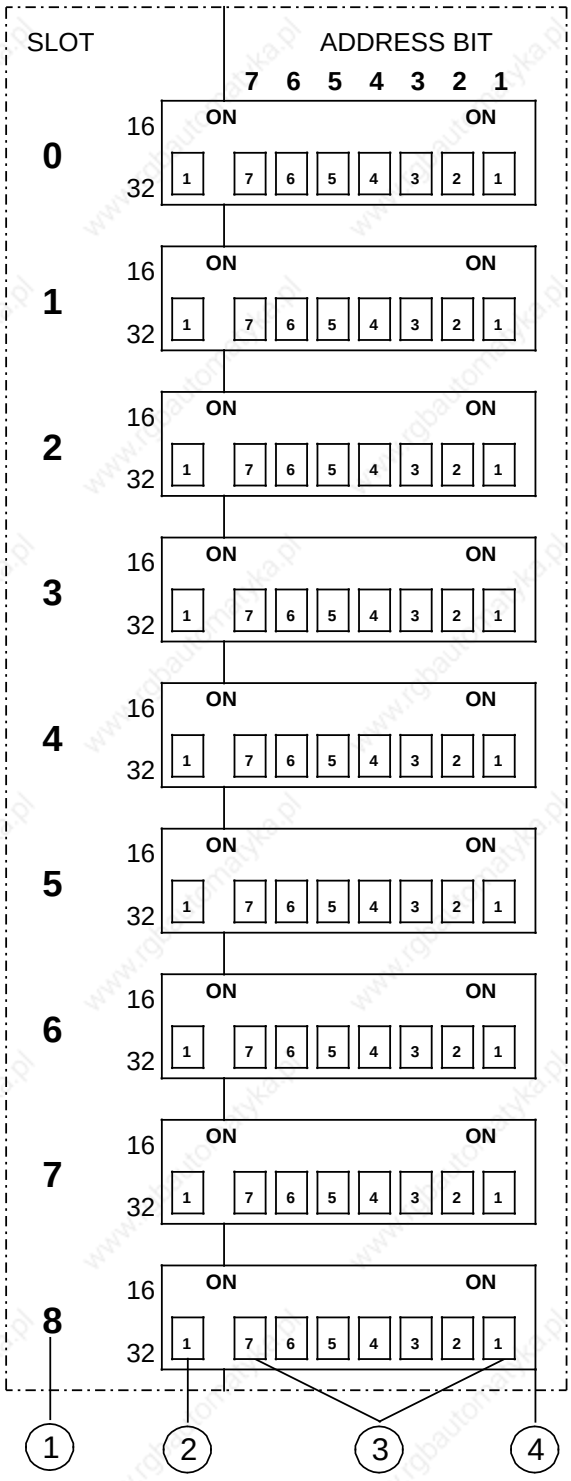
**Figure 6-3. Fixed Slot Addressing in the Expansion Unit**

## 6.2.2 Variable Slot Address Assignments

The S5-115U offers you the possibility of assigning an address to each slot. You can do this if an IM 306 interface module is plugged into the central controller and each expansion unit. For addressing purposes, it does not matter whether the module in question is plugged into a central controller or an expansion unit. Under a hinged cover on the right side of the interface module is an addressing panel. It has a DIP switch for each slot. Use the DIP switch to set the least significant byte number for a particular slot.

**Note**

Input and output modules in different slots can have the same address.




| Addresses for digital modules | Address switches (ON=1, OFF=0) 7 | 6 | 5 | 4 | 3 | 2 | 1 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 4 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 8 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 12 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 14 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 16 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 20 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| 22 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| 24 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 26 | 0 | 0 | 0 | 1 | 1 | 0 | 1 |
| 28 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 30 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 32 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 34 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 36 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 38 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 40 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 42 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 44 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| 46 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| 48 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 50 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 52 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| 54 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 56 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 58 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 60 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| 62 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |

| Addresses for analog modules | Address switches 7 | 6 | 5 | 4 | 3 | 2 | 1 |
|---|---|---|---|---|---|---|---|
| 128 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 144 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 160 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 176 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 192 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 208 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 224 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 240 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

: Slot number

: Switch for setting the number of inputs or outputs per slot

: Address switches

: DIP switch

**Figure 6-4. Setting Addresses on the Addressing Panel of the IM 306 Interface Module**

**Setting Addresses**

Use the left-hand switch ( in Figure 6-4) on the addressing panel of the IM 306 to indicate what type of module you have plugged into the slot. Proceed as follows:

Set the switch to OFF: for a 32-channel digital module or a 16-channel analog module.
Set the switch to ON: for a 16-channel digital module or an 8-channel analog module.

The following modules must also be set as 16-channel digital modules:
- 482-7 digital input/output module
- 434-7 digital input module with process interrupt.

Use the seven address switches ( in Figure 6-4) on the addressing panel of the IM 306 to indicate the least significant address (the address for channel "0") for the module in question. This setting establishes the addresses of the other channels in ascending order.

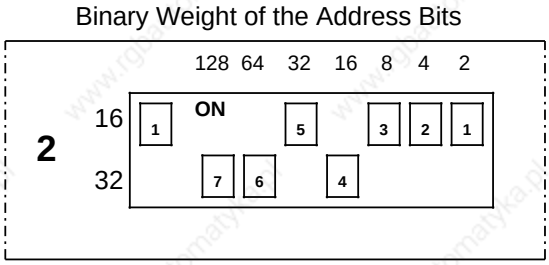
When setting start addresses, note the following:

- 32-channel digital modules can only have start addresses whose byte numbers are divisible by 4 (e.g., 0, 4, 8 ...).
- 16-channel digital modules can only have start addresses whose byte numbers are divisible by 2 (e.g., 0, 2, 4 ...).
- 16-channel analog modules can only have the start addresses 128, 160, 192 and 224.
- 8-channel analog modules can only have the start addresses 128, 144, 160 to 240.

**Example**

A 16-channel digital input module is plugged into slot 2.
Assign it start address 46.0 by performing the following steps:

- Check to see if the byte number of the start address can be divided by 2 since you are dealing with a 16-channel digital module.
  46 : 2=23 Remainder 0
- Set the number of input channels (set switch to ON).
- Set the address switches on the DIP switch for slot number 2 as shown in Figure 6-5.




The address is equal to the sum of the weights selected by the individual coding switches, e.g.:

2+4+8+32=46

**Figure 6-5. Setting a DIP Switch**

The module is then addressed as follows:

| Channel No. | 0 | 1 | 2 . . . | 7 | 8 | 9 | 10 . . . | 15 |
|---|---|---|---|---|---|---|---|---|
| Address | 46.0 | 46.1 | | 46.7 | 47.0 | 47.1 | | 47.7 |

## 6.3 Handling Process Signals

Input/output module signal states can be read from or written to the addresses shown in Figure 6-6.

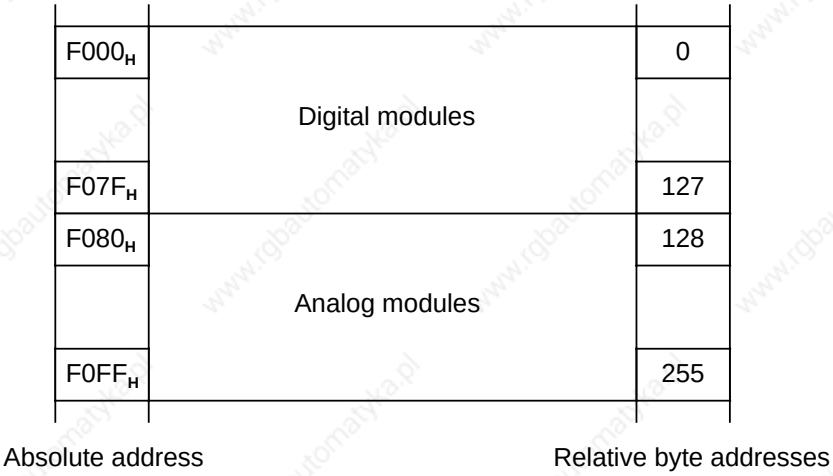



**Figure 6-6. Addresses of the Input/Output Modules**

Digital module signal states are also stored in a special memory area called the process image. The process image has two sections, namely the process input image (PII) and the process output image (PIQ). Figure 6-7 shows where the process images are located in the program memory.

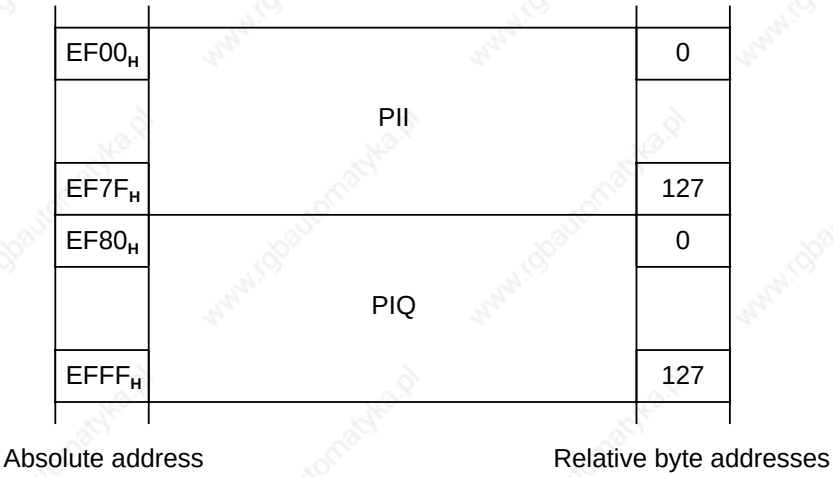



**Figure 6-7. Location of the Process Images**

Process signals can be read or output either via the process image or directly.

### 6.3.1 Accessing the PII

At the beginning of program scanning, the input module signal states are written to the PII. The statements in the control program use a particular address to indicate what information is currently needed. The control logic then reads the data that was current at the beginning of program scanning and works with it.

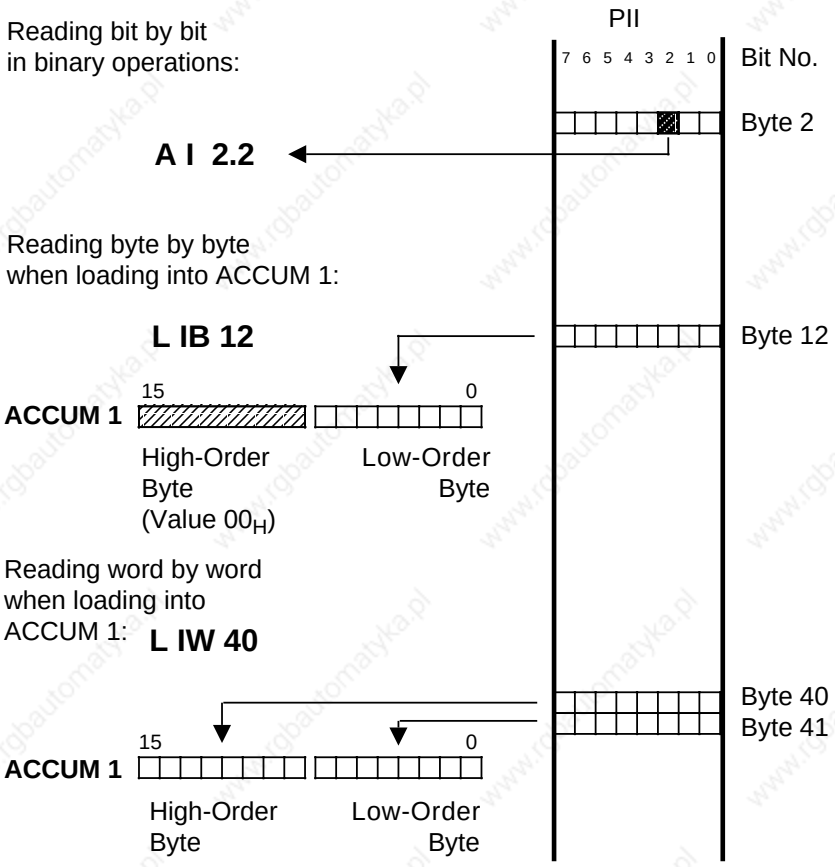



**Figure 6-8. Accessing the PII**

Reading of the PII can be inhibited.
To do so, it is necessary to act upon bit 1 of system data word SD 120 ($EAF0_H$) using system operations (Load and Transfer).

Bit 1="1": Reading of inputs is inhibited.
Bit 1="0": Reading of inputs is enabled.

The default setting is bit 1="0" (read enabled).

> **Note**
>
> System data word SD 120 can be modified with the programmer function DISPL. ADDR. only when the PLC is in the STOP mode!

### 6.3.2 Accessing the PIQ

New signal states are entered in the PIQ during program scanning. This information is transferred to the output modules at the end of each program scan.

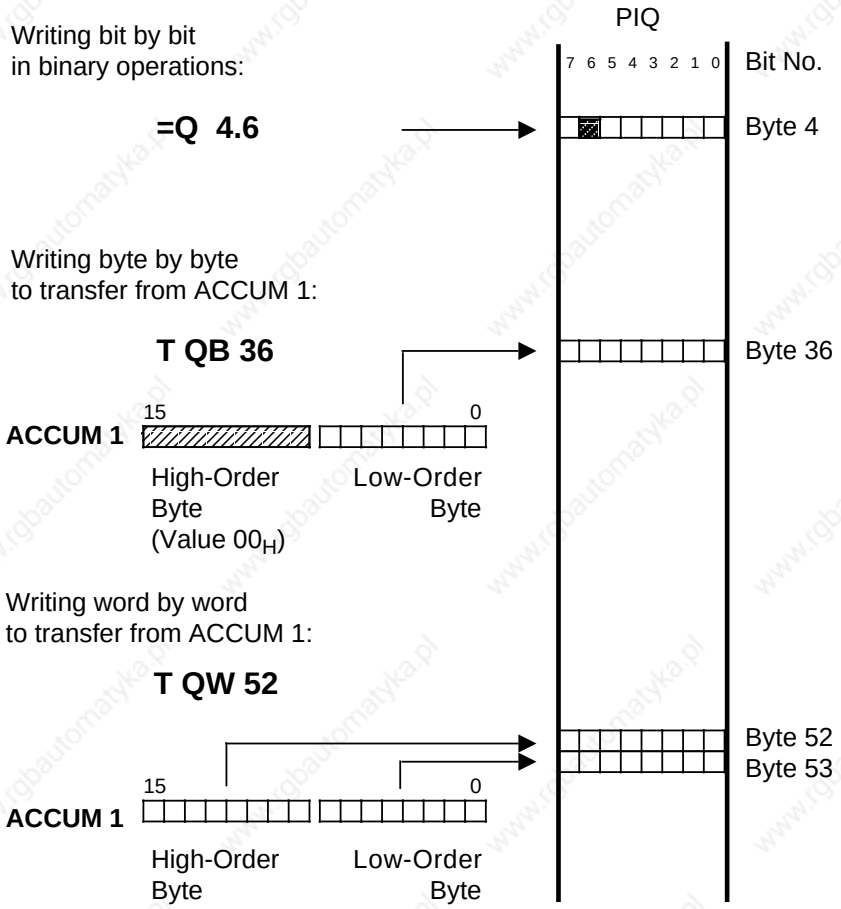



**Figure 6-9. Accessing the PIQ**

On all CPUs, output of the PIQ to the output modules can be inhibited by setting bit 2 in system data word SD 120 (EA70$_H$).

Bit 2="1": Output of the PIQ is inhibited.
Bit 2="0": Output of the PIQ is enabled.

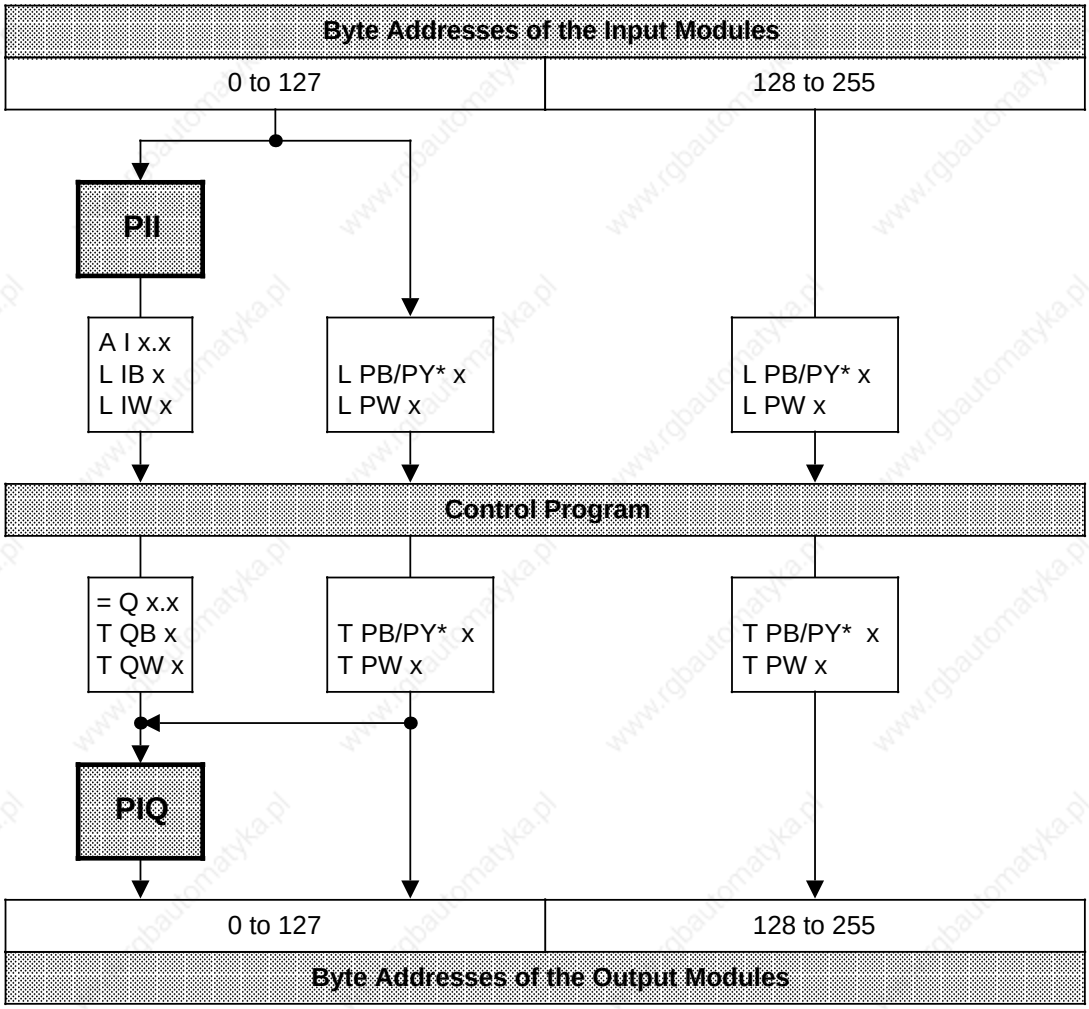
The default setting is bit 2="0" (output of the PIQ enabled).

**Note**

System data word SD 120 can be modified over the programmer function DISPL. ADDR. only when the PLC is in the STOP mode!

### 6.3.3 Direct Access

Analog module signal states are not written to the process image. They are read in or transferred to an output module directly with the "L PB/PY* x", "L PW x", "T PB/PY* x", or "T PW x" statements.
You can also exchange information with digital modules directly. This is necessary when signal states have to be processed immediately in the control program. Figure 6-10 shows differences during the loading of signal states.




***** PY on a programmer with S5-DOS-PG

**Figure 6-10. Loading Input/Output Modules**

**Note**

If you use direct access to call an address whose slot is unoccupied, the CPU enters the "STOP" mode with error code "QVZ" (time-out).

**Note**

On the CPU 944, the digital inputs can be read with OB254 and the PIQ output to the output modules with OB255, irrespective of the contents of system data word SD 120 (also see Chapter 11, "Integral Blocks").

## 6.4 Address Allocation on the Central Processing Units

The following figures show the contents of CPU RAM.

Important memory areas such as those for system data (SD), timers (T), counters (C), flags (F) and the block address list are shown in detail Figures 6-11 and 6-12.

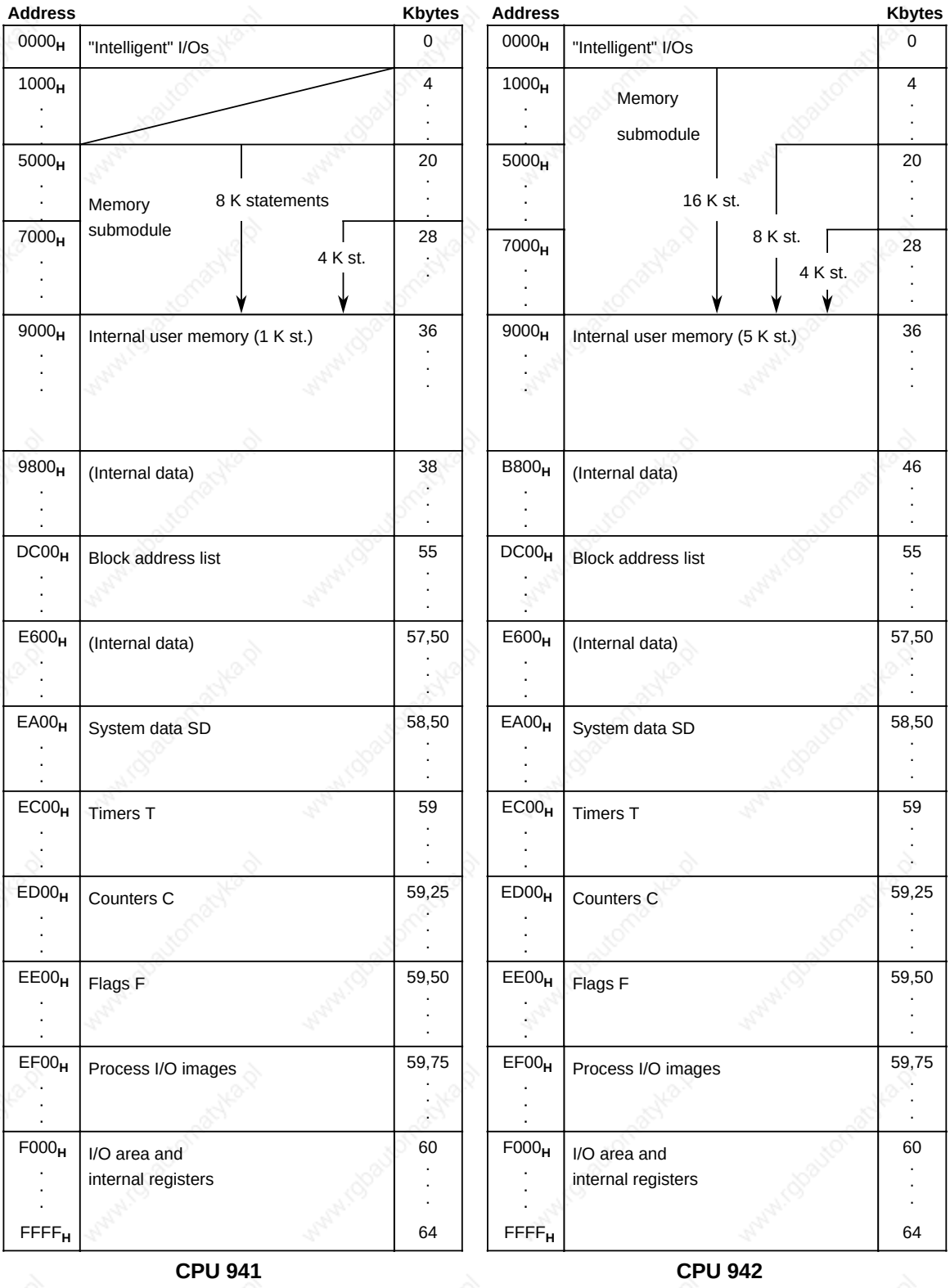



**Figure 6-11. Memory Allocation on the CPU**

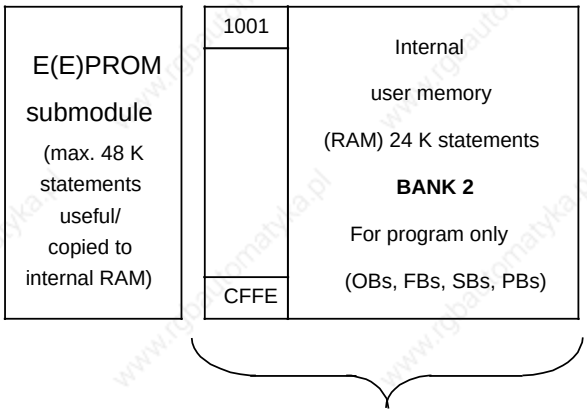
E(E)PROM submodule

(max. 24 K statements useful/ copied to internal RAM)

| Address | | Kbytes |
|---|---|---|
| $0000_H$ | "Intelligent" I/Os | 0 |
| $1000_H$ . . . | Internal user memory (RAM) 24 K statements | 4 . . . |
| $D000_H$ . . . | (Internal data) | 52 . . . |
| $DC00_H$ . . . | Block address list | 55 . . . |
| $E600_H$ . . . | (Internal data) | 57,50 . . . |
| $EA00_H$ . . . | System data SD | 58,50 . . . |
| $EC00_H$ . . . | Timers T | 59 . . . |
| $ED00_H$ . . . | Counters C | 59,25 . . . |
| $EE00_H$ . . . | Flags F | 59,50 . . . |
| $EF00_H$ . . . | Process I/O image | 59,75 . . . |
| $F000_H$ . . . $FFFF_H$ | I/O area and internal registers | 60 . . . 64 |

**CPU 943**

**Figure 6-11. Memory Allocation on the CPU (Continued)**




| Address | | Kbytes |
|---|---|---|
| $0000_H$ | "Intelligent" I/Os | 0 |
| $1000_H$ . . . | Internal user memory (RAM) 24 K statements **BANK 1** | 4 . . . |
| $D000_H$ . | (Internal data) | 52 . |
| $DC00_H$ | Block address list | 55 . . . |
| $E600_H$ . | (Internal data) | 57,50 . |
| $EA00_H$ . | System data SD | 58,50 . |
| $EC00_H$ . | Timers T | 59 . |
| $ED00_H$ . | Counters C | 59,25 . |
| $EE00_H$ . | Flags F | 59,50 . |
| $EF00_H$ . | Process I/O image | 59,75 . |
| $F000_H$ . $FFFF_H$ | I/O area and internal registers | 60 . 64 |

**CPU 944**

**Figure 6-11. Memory Allocation on the CPU (Continued)**

## Note

The block start address on bank 1 are even-numbered addresses.
The block start addresses on bank 2 are odd-numbered addresses.

The input/output area is subdivided as follows:

| Address | | Kbytes |
|---|---|---|
| $F000_H$ | I/O modules (P area) | 60 |
| $F100_H$ | Q area | 60.25 |
| $F200_H$ | Interprocessor communication flags | 60.50 |
| $F300_H$ | | 60.75 |
| $F400_H$ | Page frame | 61 |
| $F800_H$ | | 62 |
| $FC00_H$ | IM 3-area | 63.00 |
| $FD00_H$ | IM 4-area | 63.25 |
| $FEFF_H$ | Interface registers for CPs and IPs with page addressing (write-only) | |
| $FF00_H$ | (Internal registers) | 63.75 |
| $FFFF_H$ | | 64 |

**Figure 6-12. Address Assignment in the I/O Area**

The following table lists the system data of relevance to the user and indicates the sections which provide more detailed information.

**Table 6-1. Address Allocation in the System Data Area**

| System Data Word | Address (hex.) | Description | On CPU | Details in Section |
|---|---|---|---|---|
| 8 - 12 | EA10<br>:<br>EA19 | Integrated hardware clock:<br>Clock data area, status word, error codes, correction value | 943, 944 | 13 |
| 16 - 31 | EA20<br>:<br>EA3F | List of inputs and outputs (digital, analog) | 941,942, 943,944 | 2.5.2 |
| 32 | EA40<br>EA41 | Address level indicator for memory bank 2 | 944 | 5.1.3 |
| 33 | EA42<br>EA43 | Address level indicator for memory bank, for CPU 944 memory bank 1 | 941,942, 943,944 | 5.1.3 |
| 36 | EA48<br>EA49 | Start address of internal RAM | 941,942, 943,944 | |
| 37 | EA4A<br>EA4B | End address of internal RAM | 941,942, 943,944 | |
| 43 | EA56<br>EA57 | Current version ('b') | 941, 942, 943, 944 | 12 |
| 46 | EA5C<br>EA5D | Driver number and error code (e.g. for ASCII driver) | 943, 944 | 12 |
| 48 - 55 | EA60<br>:<br>EA6F | Driver parameter block (e.g. for ASCII driver) | 943, 944 | 12 |
| 57 - 63 | EA72<br>:<br>EA7F | SINEC L1 parameter block | 941,942, 943,944 | 12 |
| 64 - 79 | EA80<br>:<br>EA9F | Address list of IPC output flags | 941,942, 943,944 | 12.1.1 |
| 80 - 95 | EAA0<br>:<br>EABF | Address list of IPC input flags | 941,942, 943,944 | 12.1.1 |
| 96 | EAC0<br>EAC1 | Scan monitoring time (multiple of 10 ms) | 941,942, 943,944 | 2.6.3 |
| 97 | EAC2<br>EAC3 | Interval timer for OB13 (multiple of 10 ms) | 941,942, 943,944 | 7.4.3 |

**Table 6-1. Address Allocation in the System Data Area (Continued)**

| System Data Word | Address (hex.) | Description | On CPU | Details in Section |
|---|---|---|---|---|
| 98 | EAC4<br>EAC5 | Interval timer for OB12<br>(multiple of 10 ms) | 941,942,<br>943,944 | 7.4.3 |
| 99 | EAC6<br>EAC7 | Interval timer for OB11<br>(multiple of 10 ms) | 941,942,<br>943,944 | 7.4.3 |
| 100 | EAC8<br>EAC9 | Interval timer for OB10<br>(multiple of 10 ms) | 941,942,<br>943,944 | 7.4.3 |
| 101 | EACA<br>EACB | Time (in ms) to<br>calling OB6 | 941,942,<br>943,944 | 7.4.3 |
| 102 | EACC<br>EACD | Error codes, e.g. memory error | 943, 944 | 5.1.4 |
| 103 | EACE<br>EACF | Additional error information, e.g. absolute address of the module on time-out (QVZ) | 941, 942,<br>943,944 | 5.1.3<br>(OB23,24,27) |
| 120 | EAF0<br>EAF1 | System performance characteri-<br>stics: Software protection<br>Reading of PII inhibited<br>Output of PIQ inhibited<br>Retentivity of flags, counters and timers;<br>OB6 priority | 941, 942,<br>943, 944 | <br>11.3.7<br>6.3.1<br>6.3.2<br>4.2.3<br>7.4.4 |
| 121 | EAF2<br>EAF3 | Actual scan time | 941,942,<br>943,944 | 2.6.1 |
| 122 | EAF4<br>EAF5 | Maximum scan time | 941,942,<br>943,944 | 2.6.1 |
| 123 | EAF6<br>EAF7 | Minimum scan time | 941,942,<br>943,944 | 2.6.1 |
| 126 | EAFC<br>EAFD | Restart delay in ms | 941,942,<br>943,944 | 2.5.2 |
| 128 - 201 | EB00<br>EB93 | Block stack | 941,942,<br>943,944 | 5 |
| 202 | EB94<br>EB95 | Number of the integral FBs<br>"COMPR" and "DELETE" | 941,942,<br>943,944 | 11 |
| 203 - 238 | EB96<br>EBDD | Interrupt stack | 941,942,<br>943,944 | 5 |
| 240 - 243 | EBE0<br>EBE7 | Reserved for standard FBs | 941,942,<br>943,944 | |
| 248 - 255 | EBF0<br>EBFF | Reserved for user | 941,942,<br>943,944 | |

**Table 6-2. Address Allocation in the Flag, Timer and Counter Areas**

| Memory Area | | Details in Section | Abs. Address (hex.) |
|---|---|---|---|
| Flags (F) | FY 0<br>FY 1<br>:<br>FY 255 | 8.1.2 | EE00<br>EE01<br>:<br>EEFF |
| Timers (T) | T 0<br>T 1<br>:<br>T 127 | 8.1.4 | EC00, EC01<br>EC02, EC03<br>:<br>ECFE, ECFF |
| Counters (C) | C 0<br>C 1<br>:<br>C 127 | 8.1.5 | ED00, ED01<br>ED02, ED03<br>:<br>EDFE, EDFF |

**Table 6-3. Block Address List**

| Block Type | Block Number | Abs. Address (hex.) |
|---|---|---|
| Organization blocks | OB0<br>OB1<br>:<br>OB255 | DC00, DC01<br>DC02, DC03<br>:<br>DDFE, DDFF |
| Function blocks | FB0<br>FB1<br>:<br>FB255 | DE00, DE01<br>DE02, DE03<br>:<br>DFFE, DFFF |
| Program blocks | PB0<br>PB1<br>:<br>PB255 | E000, E001<br>E002, E003<br>:<br>E1FE, E1FF |
| Sequence blocks | SB0<br>SB1<br>:<br>SB255 | E200, E201<br>E202, E203<br>:<br>E3FE, E3FF |
| Data blocks | DB0<br>DB1<br>:<br>DB255 | E400, E401<br>E402, E403<br>:<br>E5FE, E5FF |

## 7 Introduction to STEP 5


7.1 Writing a Program . . . . . 7 - 1
7.1.1 Methods of Representation . . . . . 7 - 1
7.1.2 Operand Areas . . . . . 7 - 3
7.1.3 Circuit Diagram Conversion . . . . . 7 - 3

7.2 Program Structure . . . . . 7 - 4
7.2.1 Linear Programming . . . . . 7 - 4
7.2.2 Structured Programming . . . . . 7 - 5

7.3 Block Types . . . . . 7 - 7
7.3.1 Organization Blocks (OBs) . . . . . 7 - 8
7.3.2 Program Blocks (PBs) . . . . . 7 - 11
7.3.3 Sequence Blocks (SBs) . . . . . 7 - 11
7.3.4 Function Blocks (FBs) . . . . . 7 - 11
7.3.5 Data blocks (DBs) . . . . . 7 - 16

7.4 Program Execution . . . . . 7 - 18
7.4.1 RESTART Program Execution . . . . . 7 - 18
7.4.2 Cyclic Program Execution . . . . . 7 - 20
7.4.3 Time-Controlled Program Execution . . . . . 7 - 20
7.4.4 Interrupt-Driven Programming Execution . . . . . 7 - 22
7.4.5 Handling Programming Errors and PLC Malfunctions . . . . . 7 - 23

7.5 Processing Blocks . . . . . 7 - 25
7.5.1 Modifying the Program . . . . . 7 - 25
7.5.2 Modifying Blocks . . . . . 7 - 25
7.5.3 Compressing the Program Memory . . . . . 7 - 25

7.6 Number Representation . . . . . 7 - 26

## Figures


7-1. Compatibility of STEP 5 Methods of Representation . . . . . 7 - 2
7-2. Nesting . . . . . 7 - 6
7-3. Structure of a Block Header . . . . . 7 - 8
7-4. Example of Organization Block Use . . . . . 7 - 10
7-5. Programming a Function Block Parameter . . . . . 7 - 13
7-6. Assigning Function Block Parameters . . . . . 7 - 16
7-7. Example of the Contents of a Data Block . . . . . 7 - 17
7-8. Validity Areas of Data Blocks . . . . . 7 - 17
7-9. Determining the Order of Priority of OB6 in System Data Word 120 . . . . . 7 - 22
7-10. Compression Process . . . . . 7 - 25
7-11. Bit Assignment of a 16-Bit Fixed-Point Binary Number . . . . . 7 - 26


## Tables


7-1. Comparison of Operation Types . . . . . 7 - 2
7-2. Comparison of Block Types . . . . . 7 - 7
7-3. Overview of Organization Blocks . . . . . 7 - 9
7-4. Block Parameter Types and Data Types with Permissible Actual Operands . . . . . 7 - 14
7-5. Parameter Block for Time OBs . . . . . 7 - 21
7-6. Examples of Number Representation in the PLC . . . . . 7 - 26

# 7 Introduction to STEP 5

This chapter explains how to program the S5-115U. 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.

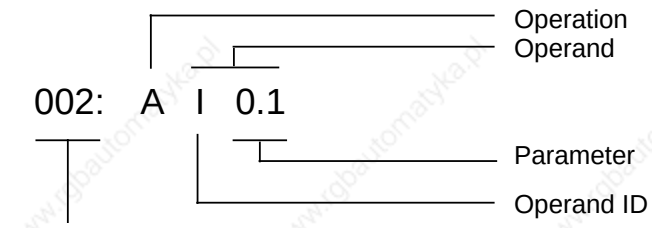
## 7.1 Writing a Program

A control program specifies a series of operations that tell the programmable controller (PLC) how it has to control a system. You must write the program in a very 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.

### 7.1.1 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.

- **Control System Flowchart (CSF)**
  CSF represents logic operations with symbols.

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

- **GRAPH 5**
  GRAPH 5 is a graphic representation of the structure of sequence controls.

The last three methods of representation indicated are only possible with CRT-based programmers (e.g. PG 635, PG 750).

Each method of representation has its special characteristics. Therefore, 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 7-1 illustrates these points in a diagram.

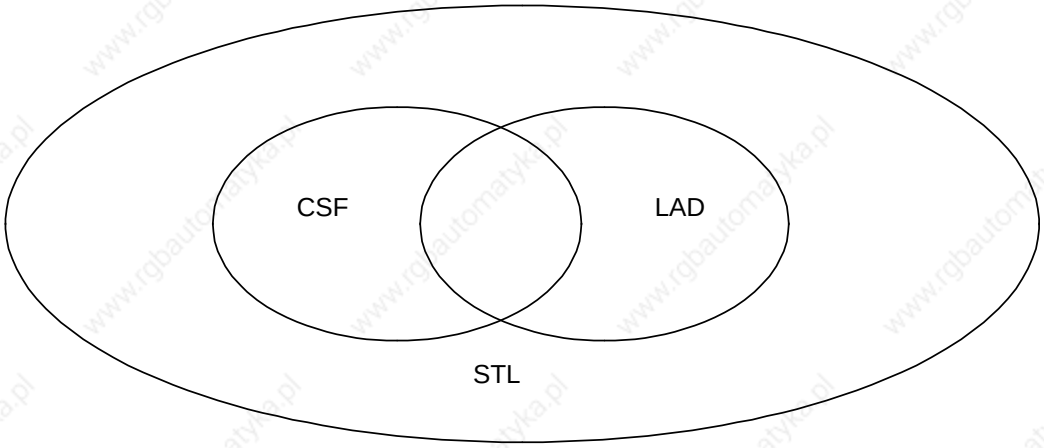



**Figure 7-1. Compatibility of STEP 5 Methods of Representation**

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

- basic
- supplementary
- system

Table 7-1 provides further information on these operations.

**Table 7-1. Comparison of Operation Types**

| STEP 5 PROGRAMMING LANGUAGE | | | |
|---|---|---|---|
| | **Basic Operations** | **Supplementary Operations** | **System Operations** |
| Application | in all blocks | only in function blocks | only in function blocks |
| Methods of Representation | STL, CSF, LAD | STL | STL |
| Special features | | | for users with good system knowledge |

Refer to Chapter 8 for a description of all operations and programming examples.

### 7.1.2 Operand Areas

The STEP 5 programming language has the following operand areas:

| | | |
|---|---|---|
| **I** | (inputs) | interfaces from the process to the PLC |
| **Q** | (outputs) | interfaces from the PLC to the process |
| **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** | (input/output modules) | interface between process and programmable controller |
| **K** | (constants) | defined numeric values |
| **OB, PB, SB, FB, DB** | (blocks) | program structuring aids |
| **BS** | (system data) | interface between operating system and control program |

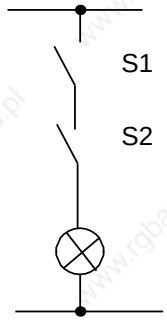
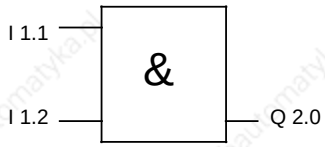
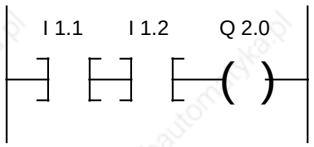
Refer to Appendix A for a listing of all operations and operands.

### 7.1.3 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 supposed 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 logic operation (RLO) is assigned to output 2.0 (the lamp lights up).

| Circuit Diagram | STL | CSF | LAD |
|---|---|---|---|
|  | A I 1.1<br>A I 1.2<br>= Q 2.0 |  |  |

## 7.2 Program Structure

An S5-115U program can be one of the two following types:
- linear
- structured

Sections 7.2.1 and 7.2.2 describe these program types.

### 7.2.1 Linear Programming

When processing simple automation tasks, it is enough to program the individual operations in one block.
In the case of the S5-115U, this is organization block 1 (OB1) (see Section 7.3.1). This block is scanned cyclically, i.e. after processing the last statement, the processor returns to the first statement.

Please note the following:
- When OB1 is called, five words are assigned to the block header (see Section 7.3.1).
- 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, OB1 must be terminated by a Block End statement (BE).

### 7.2.2 Structured Programming

To solve complex tasks, it is advisable to divide an entire program into individual, self-contained program parts (blocks). This procedure has the following advantages:

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 technical 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 32 levels (see Section 7.3).

**Note**

When calculating the nesting depth, note that the system program itself can call an organization block under certain circumstances (e.g. OB32).

The total nesting depth is the sum of the nesting depths of all programmed organization blocks. If nesting goes beyond 32 levels, the PLC goes into the "STOP" mode with the error message "STUEB" (block stack overflow) (see Section 5.1). Figure 7-2 illustrates the nesting principle.

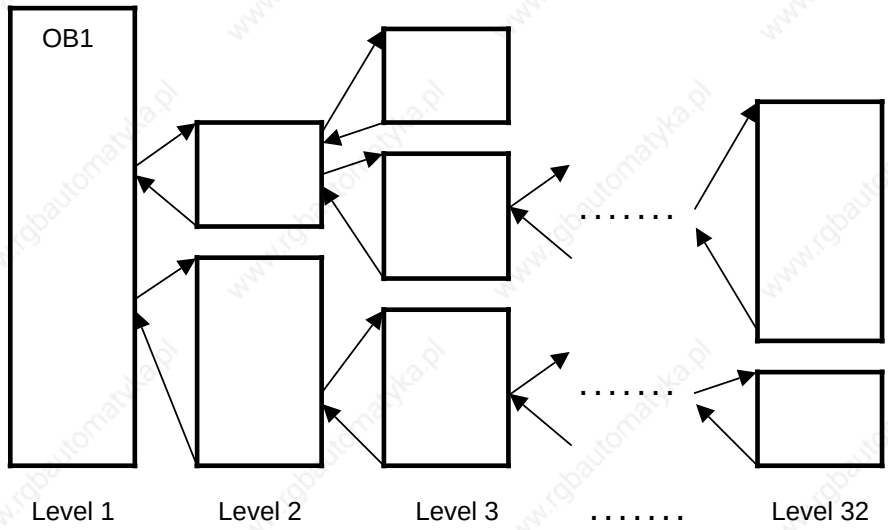



**Figure 7-2. Nesting**

## 7.3 Block Types

Table 7-2 lists the most important features of the block types.

**Table 7-2. Comparison of Block Types**

| | OB | PB | SB | FB | DB |
|---|---|---|---|---|---|
| Quantity | 256 [1] OB 0 to OB 255 | 256 PB 0 to PB 255 | 256 SB 0 to SB 255 | 256 [2] FB 0 to FB 255 | 254 [3] DB 2 to DB 255 |
| Maximum length | 8 Kbytes | 8 Kbytes | 8 Kbytes | 8 Kbytes | 2042 Data words [4] |
| Operations set (contents) | Basic operations | Basic operations | Basic operations | Basic, supple-mentary, system operations | Bit pattern<br>numbers<br>texts |
| Representa-tion methods | STL, CSF, LAD | STL, CSF, LAD | STL, CSF, LAD | STL | |
| Block header length | 5 words | 5 words | 5 words | 5 words | 5 words |

1 Organization blocks are already integrated in the operating system (see Chapter 11). Some OBs are called by the operating system autonomously (see Section 7.3.1)

2 Function blocks are already integrated in the operating system (see Section 11)

3 Data blocks DB0 and DB1 are reserved

4 Can be accessed with "L DW, L DL, L DR" or " T DW, T DR, T DL" or "TB D, TBN D, SU D, RU D" up to DW 255.

**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.

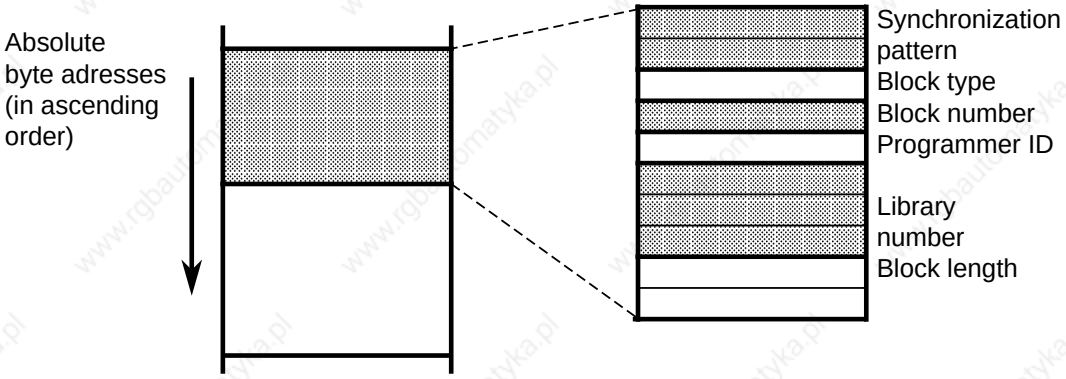



**Figure 7-3. Structure of a Block Header**

**Programming**

Blocks are programmed with the "LAD, CSF, STL" software package. See your programmer manual for details of programming individual blocks.

## 7.3.1 Organization Blocks (OBs)

Organization blocks are the interface between the operating system and the control program; they can be divided into three groups:
- An organization block is called cyclically by the operating system (OB1)
- Some organization blocks are event-driven or time-controlled; i.e. they are called by
  - STOP RUN or POWER OFF POWER ON transitions (OB21, OB22)
  - Interrupts (OB2 to OB6)
  - Programming errors or PLC faults (OB19, OB23, OB24, OB27, OB32, OB34)
  - Expiry of an interval (OB10 to OB13)
- Other organization blocks represent operating functions (similar to integral function blocks), which can be called from the control program (see Chapter 11).

**Table 7-3. Overview of Organization Blocks**

| OB No. | Function | OB integrated in CPU | | | |
|---|---|---|---|---|---|
| | | 941 | 942 | 943 | 944 |
| **OB must be user-programmed and be called by the operating system** | | | | | |
| OB1 | Cyclic program scanning | ▒ | ▒ | ▒ | ▒ |
| **OBs for interrupt-driven and time-controlled program scanning** | | | | | |
| OB2 | Interrupt A: Digital input module -434 and IP generate interrupt | ▒ | ▒ | ▒ | ▒ |
| OB3 | Interrupt B: IP generates interrupt | ▒ | ▒ | ▒ | ▒ |
| OB4 | Interrupt C: IP generates interrupt | ▒ | ▒ | ▒ | ▒ |
| OB5 | Interrupt D: IP generates interrupt | ▒ | ▒ | ▒ | ▒ |
| OB6 | Interrupt generated by internal timers | ▒ | ▒ | ▒ | ▒ |
| OB10 | Time-controlled | ▒ | ▒ | ▒ | ▒ |
| OB11 | program scanning | ▒ | ▒ | ▒ | ▒ |
| OB12 | | ▒ | ▒ | ▒ | ▒ |
| OB13 | (variable in each case: 10 msec. to 10 min.) | ▒ | ▒ | ▒ | ▒ |
| **OBs for controlling restart characteristics** | | | | | |
| OB21 | Manual switch on (STOP RUN) | ▒ | ▒ | ▒ | ▒ |
| OB22 | Automatic switch on when power is restored | ▒ | ▒ | ▒ | ▒ |
| **OBs for handling programming errors and PLC faults** | | | | | |
| OB19 | When a block is called which has not been loaded | ▒ | ▒ | ▒ | ▒ |
| OB23 | Time-out during individual access to the S5 bus (e.g. LPB, LIR, etc.) | ▒ | ▒ | ▒ | ▒ |
| OB24 | Time-out during update of the process image and the interprocessor communication flags | ▒ | ▒ | ▒ | ▒ |
| OB27 | Substitution error | ▒ | ▒ | ▒ | ▒ |
| OB32 | Transfer errors in DB or with GDB operation | ▒ | ▒ | ▒ | ▒ |
| OB34 | Battery failure | ▒ | ▒ | ▒ | ▒ |
| **OBs which offer operating functions** | | | | | |
| OB31 | Scan time triggering | ▒ | ▒ | ▒ | ▒ |
| OB160 | Programmable time loop | ▒ | ▒ | ▒ | ▒ |
| OB251 | PID algorithm | ▒ | ▒ | ▒ | ▒ |
| OB254 | Read in process I/O image | | | | ▒ |
| OB255 | Output process I/O image | | | | ▒ |

▒ OB available

Figure 7-4 shows how to set up a structured control program. It also illustrates the significance of organization blocks.

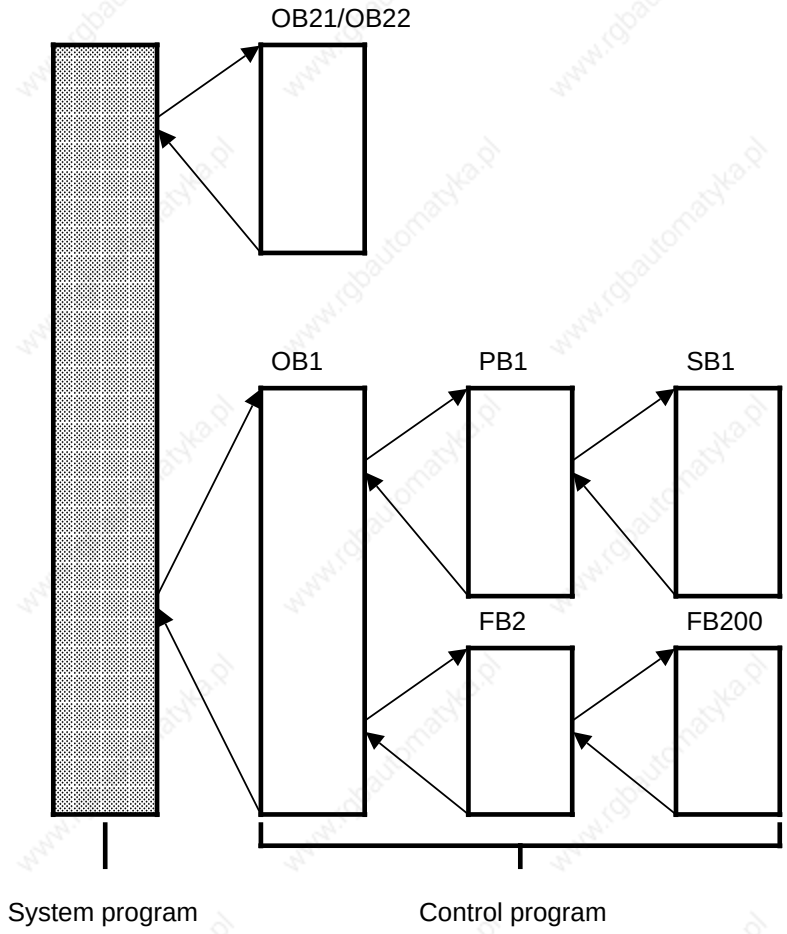



**Figure 7-4. Example of Organization Block Use**

### 7.3.2 Program Blocks (PBs)

Self-contained program parts are usually programmed in blocks.

Special feature:
Control functions can be represented graphically in program 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 be evaluated there.

### 7.3.3 Sequence Blocks (SBs)

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

### 7.3.4 Function Blocks (FBs)

Frequently recurring or complex control functions are programmed in function blocks. Function blocks have the following special features:

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 S5-115U has the following types of function blocks:
- FBs that you can program
- FBs that are integrated in the operating system (see Chapter 11)
- FBs that are available as software packages (Standard Function Blocks, see Catalog ST 57).

**Block Header**

In contrast to other types of blocks, function blocks have other organization information in addition to the block header.

Its memory requirements consist of the following:
- block description as for other blocks (five words)
- block name (five words)
- block parameter for parameter assignment (three words per parameter).

**Creating a Function Block**

In contrast to other blocks, parameters can be assigned to FBs. To assign parameters, you must program the following block parameter information:

- ***Name*** of the block parameter (formal operands)
  Each formal operand receives a name (declaration "DECL"). The name can contain up to four characters and must begin with an alpha character. You can program up to 40 block 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

Output parameters are represented to the right of the function symbol in graphics representation (CSF). The other parameters are to the left.

- Block Parameter Data ***Type***
  You can specify the following data types:
  - BI for operands with bit address
  - BY for operands with byte address
  - W for operands with word address
  - K for constants

The name, type of parameter and type of data must be entered when setting parameters.

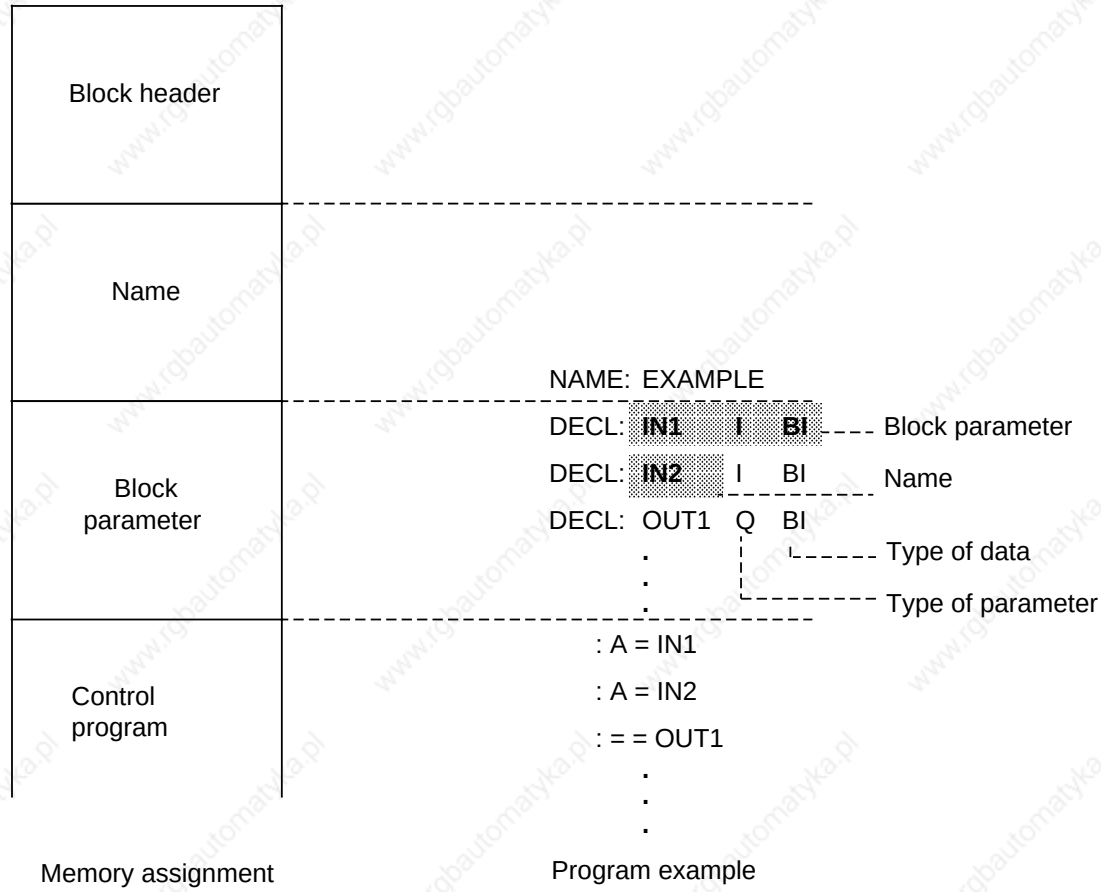



**Figure 7-5. Programming a Function Block Parameter**

**Table 7-4. 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<br>Q x.y<br>F x.y | inputs<br>outputs<br>flags |
| | BY | for an operand with byte address | IB x<br>QB x<br>FB x<br>DL x<br>DR x<br>PB x | input bytes<br>output bytes<br>flag bytes<br>data bytes left<br>data bytes right<br>peripheral bytes |
| | W | for an operand with word address | IW x<br>QW x<br>FW x<br>DW x<br>PW x | input words<br>output words<br>flag words<br>data words<br>peripheral words |
| D | KM<br>KY<br>KH<br>KS<br>KT<br>KC<br>KF | for a binary pattern (16 digits)<br>for two absolute numbers, one byte each, each in the range from 0 to 255<br>for a hexadecimal pattern (maximum 4 digits)<br>for a character (maximum 2 alphanumeric characters)<br>for a time (BCD-coded time) with time base 0.0 to 999.3<br>for a count (BCD-coded) 0 to 999<br>for a fixed-point number in the range from - 32768 to +32767 | Constants | |
| B | Type designation not permitted | | DB x<br>FB x<br>PB x<br>SB x<br>OB x | Data blocks. The C DBx operation is executed.<br>Function blocks (permissible without parameters only) are called unconditionally (JU..x).<br>Program blocks are called unconditionally (JU..x).<br>Sequence blocks are called unconditionally (JU..x).<br>Organization blocks are called unconditionally (SPA..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 | | C | Counter. The count should be assigned parameters as data or be programmed as a constant in the function block. |

### Calling a Function Block

Function blocks are stored like other blocks under a specific number (e.g. FB47) in internal program memory. Numbers 240 to 251 are reserved for integral FBs and can therefore not be used for user-written FBs! Numbers 238 to 239 are also reserved for integral FBs; however, these FBs can be renumbered (see Chapter 11).
FB calls can be programmed in all blocks except data blocks.

A function block call consists of:

- Call statement
  - JU FBx Absolute call of the FB x (***J***ump ***A***bsolute...)
  - JC FBx Call if RLO=1 (***J***ump ***C***onditional...)
- Parameter list (only necessary if block parameters have been defined in the FB)

Function blocks can only be called if they have already been programmed. When a function block call is being programmed, the programmer automatically requests the parameter list for the FB, provided block parameters have been defined in the FB.

**Assigning Function Block Parameters**

The program in the function block specifies how the formal operands (the parameters defined as "DECL") are to be processed.
As soon as you program a call statement (e.g. JU FB2), the programmer displays the ***parameter list***.
The parameter list consists of the names of the parameters each followed by a colon (:). Actual operands must now be assigned to the parameters. When the FB is called, the actual operands replace the formal operands defined in the FB so that the FB "actually" works with actual operands.
The parameter list may contain up to 40 parameters.
Example:
The name (DECL) of a parameter is IN 1, the parameter type is I (input) and the data type is BI (bit).
The formal operand of the FB then has the form
DECL: IN1 I BI.
The parameter list in the calling block specifies which (actual) operand is to replace the formal operand in the event of the FB being called; in the example this is the operand "I 1.0".
The parameter list must therefore contain the entry
IN1: I 1.0.
When the FB is called, it replaces the formal operand "EIN1" with the actual operand "I 1.0".

Figure 7-6 contains a detailed example of the parameter assignment of a function block.

The function block call occupies two words in the internal program memory and each parameter a further memory word.
The required memory length of the standard function blocks as well as the execution time are specified in Catalog ST 57.
The identifiers for the inputs and outputs of the function block appearing on the programmer during programming are deposited, together with the name, in the function block itself. It is therefore necessary to transfer all required function blocks to the program diskette (in the case of off-line programming) or enter them directly into the program memory of the PLC before programming begins on the programming unit.

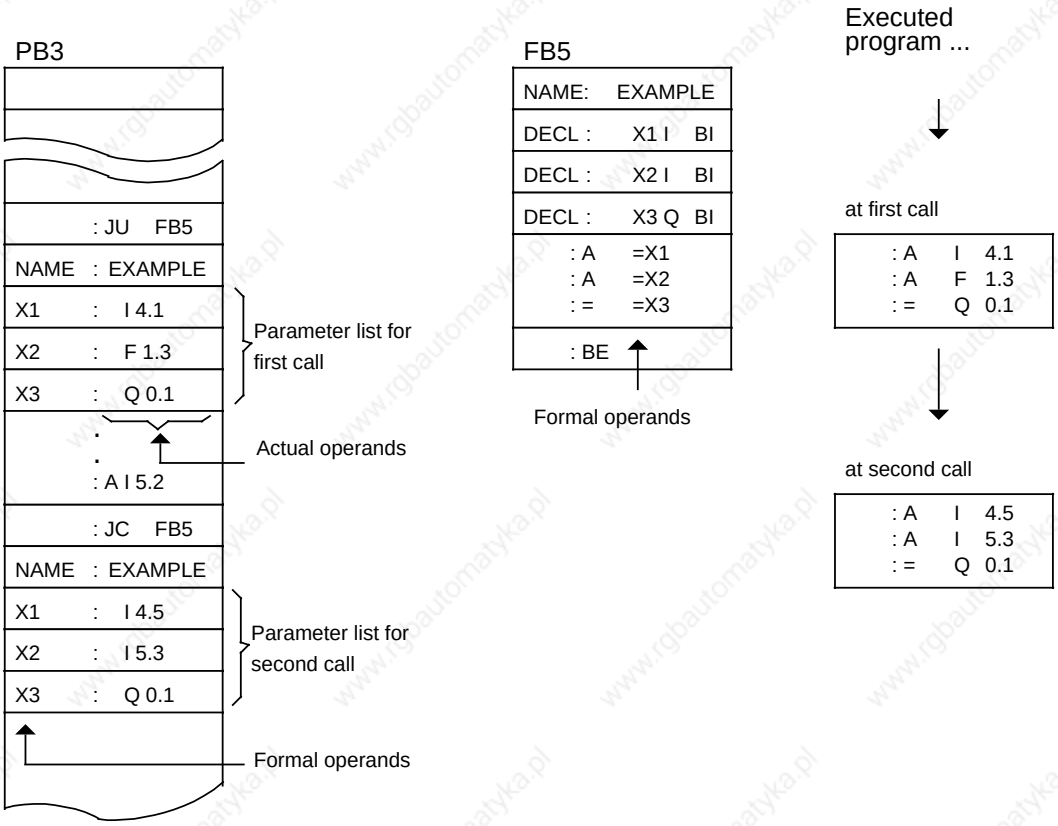



**Figure 7-6. Assigning Function Block Parameters**

## 7.3.5 Data Blocks (DBs)

Data blocks store data to be processed in a program.

The following data types are permissible:

- bit pattern (representation of controlled system states)
- hexadecimal, binary or decimal numbers (times, results of arithmetic operations)
- alphanumeric characters (message texts)

**Programming Data Blocks**

Begin data block programming by specifying a block number between 2 and 255. DB0 is reserved (for the operating system) and DB1 is reserved (for initializing internal functions (see Chapter 11) and for defining interprocessor communication flags (see Chapter 12)). The data is deposited in this block in words. If the information is less than 16 bits in volume, the higher-order bits are filled with zeros. Entry of the data begins at data word zero and is continued in ascending order. The data block can accommodate up to 2042 data words. Accessing is possible up to DW 255 using the "L DW" and "T DW" operations. Data words 256 to 2042 can only be accessed using the "LIR", "TIR" and "TNB" operations.

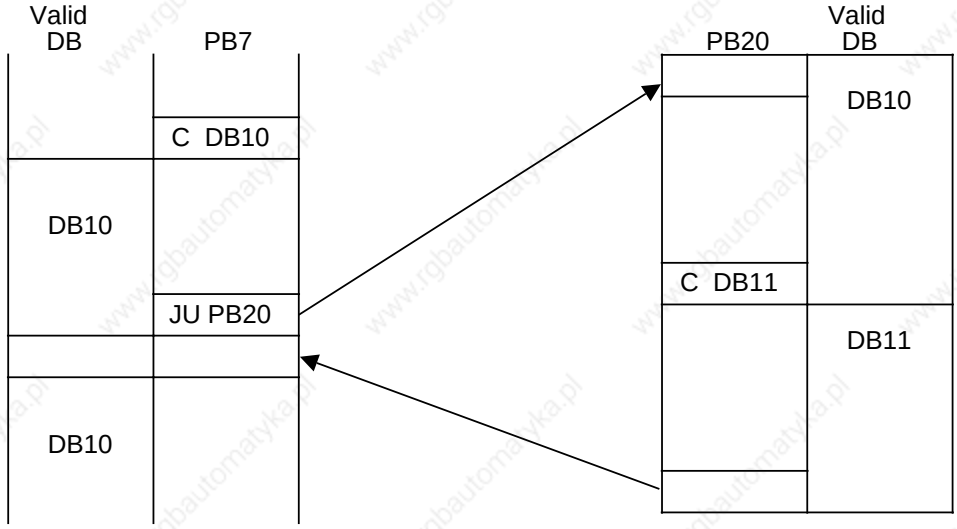
| Input | | | | Values stored | |
|---|---|---|---|---|---|
| 0000 | : | KH = | A13C | DW0 | A13C |
| 0001 | : | KT = | 100.2 | DW1 | 2100 |
| 0003 | : | KF = | +21874 | DW2 | 5572 |

**Figure 7-7. Example of the Contents of a Data Block**

Data blocks can also be generated or deleted in the control program (see Section 8.1.8).

**Program Processing with Data Blocks:**

- A data block must be called in the program with the C DB x operation (x=no.).
- Within a block, a data block remains valid until another data block is called.
- When the program jumps back into the higher-level block, the data block that was valid before the block call is again valid.
- In all organization blocks (OBs), the data blocks used by the user program must be opened with the relevant C DBxx operation.




When PB20 is called, the valid data area is entered into a memory.
When the program jumps back, this area is reopened.

**Figure 7-8. Validity Areas of Data Blocks**

## 7.4 Program Execution

Some organization blocks handle the task of structuring and managing the control program.
These OBs can be grouped as follows according to task:

- OBs for RESTART program execution
- OB for cyclic program execution
- OBs for time-controlled program execution
- OBs for (process) interrupt-driven program execution
- OBs for handling programming errors and PLC malfunctions.

There are also OBs which offer functions similar to those of the integral function blocks (e.g. PID control algorithm). These OBs are described in the chapter on "Integral Blocks" (see Chapter 11).
You will find a summary of all OBs in Section 7.3.1.

The following subsections indicate which special organization blocks are provided by the CPU for the tasks listed above and the points to watch when programming them.

### 7.4.1 RESTART Program Execution

At restart, i.e.

- after a STOP RUN transition (manual cold restart),
  and
- after a POWER OFF POWER ON transition (automatic cold restart after power restore, if the CPU was previously in the RUN mode),

the operating system of the CPU automatically calls a RESTART OB; provided one is programmed:

- OB21 (in the case of manual cold restart)
  or
- OB22 (in the case of automatic cold restart after power restore, if the CPU was previously in the RUN mode).

If you have programmed these blocks, this program is executed before cyclic program execution; it is therefore suitable for, e.g. (one-off) setting of specific system data. If the relevant RESTART OB is not programmed, the CPU jumps direct to the RUN mode (cyclic program execution, OB1).
The restart characteristics of the CPUs are detailed in Section 2.5.2. Here we will give examples of how a RESTART OB can be programmed.

**Example 1:** Programming OB22

After power restoration, check that all input/output modules are ready for operation. If one or more modules cannot be accessed (are not plugged in or have malfunctioned), the PLC should jump to the STOP mode.

| OB22 STL | Explanation |
|---|---|
| :L KF +0<br>:T PW 0<br>:T PW 2<br>:T PW 4<br>:L PW 6<br>:L PW 8<br>:L PW 10<br>:BE | Output words 0, 2, and 4 are set to "0".<br><br>The information in input words 6, 8, and 10 is loaded into ACCUM 1 consecutively. |

If an input/output module cannot be accessed with the statement L PW or T PW, the CPU jumps to the STOP mode at this statement and the interrupt bit QVZ (timeout) is set in the ISTACK (see Section 5.1).

**Example 2:** Programming OB21 and FB1

After cold restart with the mode selector, flag bytes 0 to 99 are supposed to be preset with "0". Flag bytes 100 to 127 are supposed to be retained since they contain important machine information.

**Prerequisite:** The retentive switch must be on the retentive setting (RE).

| OB21 STL | Explanation |
|---|---|
| :JU FB 1<br>NAME :CLEAR F<br>:BE | Call FB 1 unconditionally. |

| FB1 STL | | | | Explanation |
|---|---|---|---|---|
| NAME | :CLEAR F | | | |
| | :L | KF | +0 | FW 200 is preset with "0". |
| | :T | FW | 200 | "0" is stored in ACCUM 1. |
| L10 | :L | KF | +0 | The contents of FW 200 indicate the address of the |
| | :DO | FW | 200 | current flag word. |
| | :T | FW | 0 | The current flag word is set to "0". |
| | :L | FW | 200 | |
| | :I | 2 | | The contents of FW 200 are incremented by 2. |
| | :T | FW | 200 | |
| | :L | KF | +100 | The comparison value "100" is loaded into ACCUM 1. |
| | :<F | | | If the contents of FW 200 < 100, the program jumps to |
| | :JC | =L10 | | label 10. |
| | :BE | | | Bytes FB 0...99 are set to "0". |

### 7.4.2 Cyclic Program Execution

OB1 is called cyclically by the operating system. The maximum duration of the cyclic program is established by the scan monitoring time (see Section 2.6.3). If you want structured programming, you should program only jump operations (block calls) in OB1. The blocks called (PBs, FBs and SBs) should then consist of self-contained function units so that overall clarity is enhanced.
See Section 2.5.3 for details on cyclic program execution (RUN mode).

### 7.4.3 Time-Controlled Program Execution

OBs 10 to 13 are available for time-controlled program execution. The timed-interrupt OBs are called by the operating system at fixed intervals.
The call interval can be set (e.g. in the RESTART OB) in the system data as a multiple of 10 ms. It can be changed during cyclic program execution. The default call interval for OB13 is 100 ms. Call intervals from 10 ms to 10 min. can be set in system data words 97 to 100 (range: 0 to FFFF, see Table 7-5).
You can also initialize the call intervals in DB1 (see Section 11.3).
The operating system only calls a timed-interrupt OB in the following cases:

- if a call interval is >0,
  and
- if the relevant time-interrupt OB has been programmed!

Timed-interrupt OBs interrupt the cyclic program after every STEP 5 operation. Timed-interrupt OBs ***cannot*** interrupt the following:

- Integral function blocks
- OB6
- Process interrupts (OB2 to 5).

Timed-interrupt OBs themselves can be interrupted by OB6 or by process interrupts (OB2 to 5)! Please note that the call intervals may vary as a result.

The order of priority of the timed-interrupt OBs is as follows:

| | |
|---|---|
| Highest priority: | OB13 |
| | OB12 |
| | OB11 |
| Lowest priority: | OB10. |

Please note the following also:

- The "IA" operation can be used to disable the calling of all timed-interrupt OBs and this can be enabled again with the "RA" operation. A call request can be stored while the call itself is disabled.
- If timed-interrupt OBs are to be processed in the RESTART OB (OB21, OB22), you must enable interrupts with "RA" in the RESTART OB
- The block nesting depth of 32 levels must not be exceeded even when processing a time-controlled OB.
- If a time-controlled OB used "scratchflags" which are also used in the cyclic control program, the "scratchflags" must be saved in a data block while the timed-interrupt OB executes.

**Table 7-5. Parameter Block for Time OBs**

| System Data Word | Absolute Address | High Byte | Low Byte | Default |
|---|---|---|---|---|
| SD 97 | EAC2 | Time interval for OB13 | (0 to $FFFF_H$·10 ms) | 10 (=100 ms) |
| SD 98 | EAC4 | Time interval for OB12 | (0 to $FFFF_H$·10 ms) | 0 (=no call) |
| SD 99 | EAC6 | Time interval for OB11 | (0 to $FFFF_H$·10 ms) | 0 (=no call) |
| SD 100 | EAC8 | Time interval for OB10 | (0 to $FFFF_H$·10 ms) | 0 (=no call) |

**Setting an Interval Time of 1 sec. for OB13:**

```
OB 21
        :   JU   FB 21
NAME    :   TIME ON
        .
        .
```

```
OB 22
        :JU   FB 21
NAME    :TIME ON
        .
        .
```

```
FB 21

NAME    :TIME ON
        :L  KF +100
        :T  BS 97
        :BE
```

### 7.4.4 Interrupt-Driven Programming Execution

OBs 2 to 5 are called automatically by the operating system when a (process) interrupt (interrupt A, B, C or D) occurs. See Chapter 9 for more detailed information on interrupt processing.

**(Interrupt) Response After Time Expires**

OB6 has a special position. OB6 is called by the operating system when a time programmed in system data word 101 ($EACA_H$) has expired (provided interrupts are not disabled by the "RA" operation).
In OB6, you program the response after expiry of the programmed time ("timed interrupt"). Start the time by making an entry in system data word 101 ($EACA_H$) exclusively with the T RS 101 operation.

**Example:**
You have programmed OB6 with a timed response. OB6 is to be called 22 ms after starting the clock prompt. Select and start the clock prompt with the operations

```
L KF +22
T BS 101.
```

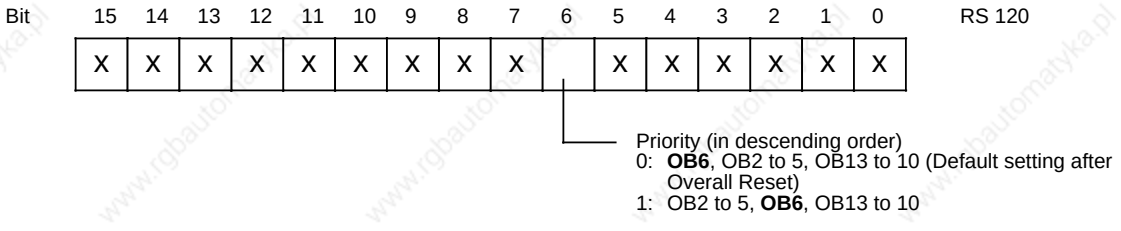
After 22 ms, OB6 interrupts the current cyclic or time-controlled program.

> **Note**
>
> A running clock prompt can be restarted if you enter another value in system data word 101. The operating system restarts the clock prompt specified by the value in ACCUM 1. A running clock prompt can be stopped (prevents OB6 call!) by transferring the value "0" to system data word 101.

After starting the clock prompt, system data word 101 contains the selected time until OB6 is called. When the programmed time has expired, the operating system enters the value "0" in system data word 101 and calls OB6.

The following applies for OB6:

- To start the clock prompt, a number (in the range 3 to 65535 or $3_H$ to $FFFF_H$) must always be transferred to system data word 101 ($EACA_H$).
- The clock prompt can be programmed in steps of 1 ms, putting the programmable clock prompt in the range 3 to 65535 ms.
- The order of priority of OB6 can be programmed in DB1 (see Section 11.3) or in system data word 120:




x = Bits which determine system characteristics
(Must not be changed when programming the priority of OB6!)

**Figure 7-9. Determining the Order of Priority of OB6 in System Data Word 120**

- OB6 cannot itself be interrupted.
- OB6 can interrupt the cyclic or time-controlled program but not a running interrupt program (OB2 to 5)! If the clock prompt expires while an interrupt OB is being processed, the OB6 call is delayed as a result.
- The OB6 call can also be delayed in the following circumstances:
  - If integral FBs are used
  - If the integral clock has been initialized
  - If PG/OP functions are pending
  - If SINEC L1 is connected
  - If a computer link or ASCII driver have been activated
    or
  - If time-controlled OBs have been programmed.

  Table 9-1 in Chapter 9 (Interrupt Processing) contains the time by which the OB6 call is delayed.

### 7.4.5 Handling Programming Errors and PLC Malfunctions

It is possible to determine the response of the CPU to errors and malfunctions using error response OBs.
The operation which triggers the timeout, substitution or transfer errors can be replaced by calling the relevant error response OB. Appropriate responses to errors can be programmed in these OBs. If only "BE" is programmed there, no response follows, i.e. the PLC does ***not*** go to STOP. If no relevant OB exists, the CPU jumps to the STOP mode.

**OB19 Response when an unloaded block is called**

In OB19, you can program the response of the CPU when an unloaded block is called.

**Example:**

The CPU is to go to STOP when an unloaded block is called:

| OB19 STL | Explanation |
|---|---|
| :STP<br>:BE | STOP statement |

If OB19 has not been programmed, the control program continues execution (no response) immediately after the jump statement (the destination of the jump does not exist!)

**OB23 Response to timeout in the case of direct I/O access**

The following operations can result in timeout: L PB; LPW; T PB; T PW; LIR; TIR; TNB.
The timeout (QVZ) error occurs if a module fails to acknowledge within 160 µs of being accessed. The cause may be a program error, a defect in the module or the removal of the module during the RUN mode. The operating system stores the absolute module address at which the QVZ occurred in system data word 103 ($EACE_H$) and calls OB23. If OB23 does not exist, the CPU goes to STOP with QVZ.

**OB24 Response to timeout when updating the process I/O image or the interprocessor communication flag**

If a timeout occurs during updating of the process I/O or the interprocessor communication flag, the absolute module address is stored in system data word 103 ($EACE_H$) and OB24 is called. If OB24 does not exist, the CPU goes to STOP with QVZ.

**OB27 Response to substitution errors**

A substitution error (SUF) can occur when the formal parameters of a function block are changed after the block is called ("JU FBx", "JC FBx").

The operating system interrupts the control program when a substitution error has been detected and then executes OB27 instead of the substitution operation. If OB27 does not exist, the CPU goes to STOP with the ISTACK error code "SUF".

**OB32 Response to transfer errors**

A transfer error (TRAF) occurs under the following circumstances:

- When data words are accessed without previously calling a data block (C DB).
- If the parameter is longer than the data block opened in the case of the operations L DW; T DW; TBD; TBN D; SU D; RU D; etc.
- If there is not sufficient user memory available to generate the specified data block in the case of the G DB (Generate Data Block).

Response to transfer error: The operating system interrupts processing of the operation where the transfer error occurred and processes OB32 instead. If OB32 does not exist, the CPU goes to STOP with the ISTACK error code "TRAF".

**OB34 Response to the BAU (battery failure) signal**

The PLC continuously checks the status of the battery in the power supply. If battery failure (BAU) occurs, OB34 is processed before each cycle until the battery has been replaced and the battery failure indicator on the power supply has been acknowledged (RESET key). You program the repsonse to battery failure in OB34. If OB34 has not been programmed, no response results.

Additional OBs which offer you operating functions, are described in Chapter 11;

OB160 Time loop
OB251 PID control algorithm
OB254 Reading in digital inputs (only in the case of CPU 944)
OB255 Outputting the process I/O image of the outputs (PIQ) to the outputs (only in the case of CPU 944)

## 7.5 Processing Blocks

Section 8 describes how to use blocks. In addition, Chapter 7 describes all operations necessary to work with blocks.

Of course, blocks that have already been programmed can be changed. Possibilities for changing blocks are described here only briefly. The operating instructions for the particular programmer used explain the necessary steps in detail.

### 7.5.1 Modifying the Program

You can modify the program, regardless of block type, with the following programmer functions:
- INPUT
- OUTPUT
- STATUS (see Chapter 4)

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

### 7.5.2 Modifying Blocks

Program modifications relate to the contents of a block. You can also delete or overwrite entire blocks. However, this does not delete the blocks in the program memory. Instead, it simply invalidates the blocks. The memory locations of these blocks cannot be written to again. As a result, new blocks might not be accepted. The programmer reports the error message "No memory space". Eliminate this by compressing the PLC memory.

### 7.5.3 Compressing the Program Memory

Figure 7-10 explains compressing. Internally one block is shifted per cycle.

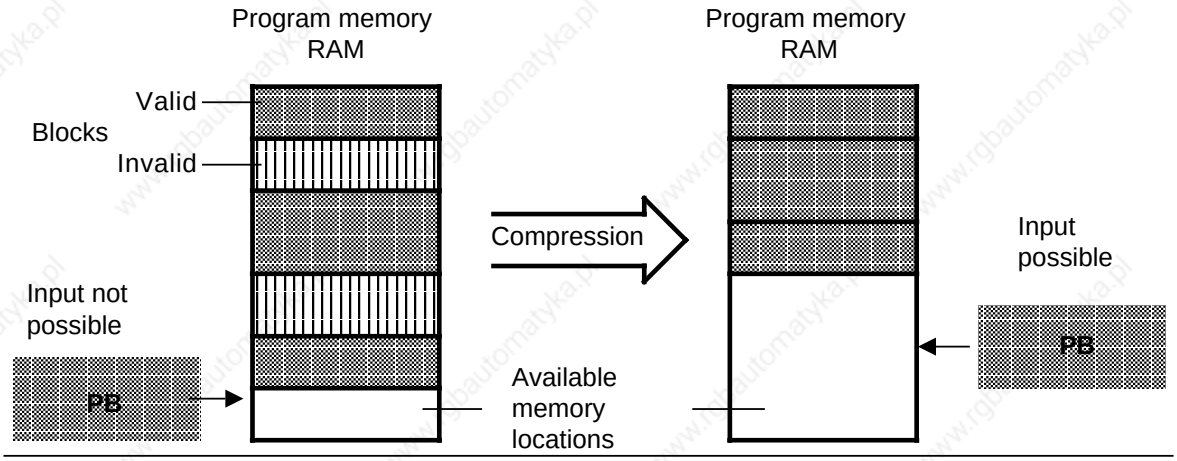



**Figure 7-10. Compression Process**

You can compress the internal program memory in the following ways:

- With the COMPRESS programmer function
  or
- With the integral FB238 (COMPR, see Chapter 11).

If a power failure occurs when shifting a block during compressing and the block shift cannot be completed, the CPU remains in the STOP mode with the error message NINEU. The bits BSTSCH, SCHTAE and SPABBR are set next to NINEU in the ISTACK.
Remedy: Overall Reset!

## 7.6 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)
- BCD-coded numbers (4 tetrads) from 0000 to 9999
- bit patterns (KM)
- constant byte (KY) from 0.0 to 255, 255

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

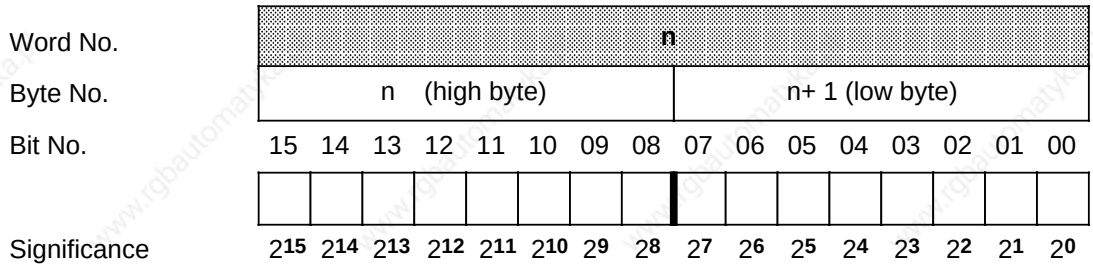



**Figure 7-11. Bit Assignment of a 16-Bit Fixed-Point Binary Number**

Table 7-6 shows three examples of number representation in the PLC.

**Table 7-6. Examples of Number Representation in the PLC**

| Entered Value | Representation in the PLC |
|---|---|
| KF - 50 | 1111 1111 1100 1110 |
| KH A03F | 1010 0000 0011 1111 |
| KY 3,10 | 0000 0011 0000 1010 |

## 8 STEP 5 Operations


8.1 Basic Operations . . . . . 8 - 1
8.1.1 Boolean Logic Operations . . . . . 8 - 2
8.1.2 Set/Reset Operations . . . . . 8 - 7
8.1.3 Load and Transfer Operations . . . . . 8 - 10
8.1.4 Timer Operations . . . . . 8 - 15
8.1.5 Counter Operations . . . . . 8 - 25
8.1.6 Comparison Operations . . . . . 8 - 30
8.1.7 Arithmetic Operations . . . . . 8 - 31
8.1.8 Block Call Operations . . . . . 8 - 32
8.1.9 Other Operations . . . . . 8 - 38

8.2 Supplementary Operations . . . . . 8 - 39
8.2.1 Load Operation . . . . . 8 - 40
8.2.2 Enable Operation . . . . . 8 - 41
8.2.3 Bit Test Operations . . . . . 8 - 42
8.2.4 Digital Logic Operations . . . . . 8 - 44
8.2.5 Shift Operations . . . . . 8 - 48
8.2.6 Conversion Operations . . . . . 8 - 50
8.2.7 Decrement/Increment . . . . . 8 - 52
8.2.8 Disable/Enable Interrupt . . . . . 8 - 53
8.2.9 Processing Operation . . . . . 8 - 54
8.2.10 Jump Operations . . . . . 8 - 57
8.2.11 Substitution Operations . . . . . 8 - 59

8.3 System Operations . . . . . 8 - 65
8.3.1 Set Operations . . . . . 8 - 65
8.3.2 Load and Transfer Operations . . . . . 8 - 66
8.3.3 Jump Operation . . . . . 8 - 69
8.3.4 Arithmetic Operation . . . . . 8 - 70
8.3.5 Other Operations . . . . . 8 - 71

8.4 Condition Code Generation . . . . . 8 - 73

8.5 Sample Programs . . . . . 8 - 76
8.5.1 Momentary-Contact Relay (Edge Evaluation) . . . . . 8 - 76
8.5.2 Binary Scaler . . . . . 8 - 77
8.5.3 Clock (Clock-Pulse Generator) . . . . . 8 - 78
8.5.4 Delay Times . . . . . 8 - 79

## Figures


8-1. Accumulator Structure . . . . . 8 - 10
8-2. Execution of the Load Operation . . . . . 8 - 12
8-3. Transferring a Byte . . . . . 8 - 12
8-4. Output of the Current Time (Example) . . . . . 8 - 18
8-5. Outputting the Current Counter Status . . . . . 8 - 27
8-6. Effect of the Processing Operation . . . . . 8 - 55


## Tables


8-1. Overview of Boolean Logic Operations . . . . . 8 - 2
8-2. Overview of the Set/Reset Operations . . . . . 8 - 7
8-3. Overview of Load and Transfer Operations . . . . . 8 - 11
8-4. Overview of Timer Operations . . . . . 8 - 15
8-5. Overview of Counter Operations . . . . . 8 - 25
8-6. Overview of Comparison Operations . . . . . 8 - 30
8-7. Overview of Arithmetic Operations . . . . . 8 - 31
8-8. Overview of Block Call Operations . . . . . 8 - 33
8-9. Other Operations . . . . . 8 - 38
8-10. Load Operation . . . . . 8 - 40
8-11. Enable Operation . . . . . 8 - 41
8-12. Overview of Bit Operations . . . . . 8 - 42
8-13. Effect of "TB" and "TBN" on the RLO . . . . . 8 - 42
8-14. Overview of Digital Logic Operations . . . . . 8 - 44
8-15. Overview of Shift Operations . . . . . 8 - 48
8-16. Overview of Conversion Operations . . . . . 8 - 50
8-17. Decrement/Increment Operations . . . . . 8 - 52
8-18. Disable/Enable Interrupt Operations . . . . . 8 - 53
8-19. Processing Operations . . . . . 8 - 54
8-20. Overview of Jump Operations . . . . . 8 - 57
8-21. Overview of Binary Logic Operations . . . . . 8 - 59
8-22. Overview of Set/Reset Operations . . . . . 8 - 60
8-23. Overview of Load and Transfer Operations . . . . . 8 - 61
8-24. Overview of Timer and Counter Operations . . . . . 8 - 62
8-25. Processing Operation . . . . . 8 - 64
8-26. Overview of Set Operations . . . . . 8 - 65
8-27. Overview of Load and Transfer Operations . . . . . 8 - 66
8-28. "JUR" Operation . . . . . 8 - 69
8-29. Arithmetic Operation . . . . . 8 - 70
8-30. Processing Operation . . . . . 8 - 71
8-31. "TAK" and "STS" Operations . . . . . 8 - 72
8-32. Condition Code Settings for Comparison Operations . . . . . 8 - 73
8-33. Condition Code Settings for Fixed-Point Arithmetic Operations . . . . . 8 - 74
8-34. Condition Code Settings for Digital Logic Operations . . . . . 8 - 74
8-35. Condition Code Settings for Shift Operations . . . . . 8 - 75
8-36. Condition Code Settings for Conversion Operations . . . . . 8 - 75

# 8 STEP 5 Operations

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

- Basic Operations include functions that can be executed in organization, program, sequence, and function blocks. Except for the addition (+F), subtraction (-F), and organizational operations, the basic operations can be input and output in the statement list (STL), control system flowchart (CSF), or ladder diagram (LAD) methods of representation.

- Supplementary Operations include complex functions such as substitution statements, test functions, and shift and conversion operations.
  They can be input and output in STL form only.

- System Operations access the operating system directly. Only an experienced programmer should use them.
  System operations can be input and output in STL form only.

## 8.1 Basic Operations

Sections 8.1.1 through 8.1.9 describe the basic operations.

### 8.1.1 Boolean Logic Operations

Table 8-1 provides an overview of boolean logic operations. Examples follow the table.

**Table 8-1. Overview of Boolean Logic Operations**

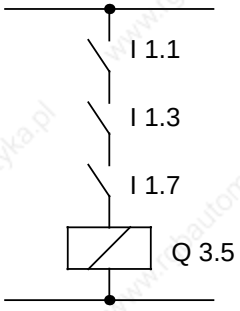
| Operation | Operand | | Meaning |
|---|---|---|---|
| **O** | | | **Combine AND operations through logic OR.**<br>Combine the result of the next AND logic operation (RLO) with the previous RLO through logic OR. |
| **A(** | | | **Combine expression enclosed in parentheses through logic AND.**<br>Combine the RLO of the expression enclosed in parentheses with the previous RLO through logic AND. |
| **O(** | | | **Combine expression enclosed in parentheses through logic OR.**<br>Combine the RLO of the expression enclosed in parentheses with the previous RLO through logic OR. |
| **)** | | | **Close parenthesis.**<br>Conclude the expression enclosed in parentheses. |
| **A** | | | **Scan operand for "1" and combine with RLO through logic AND.**<br>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. **1** |
| **O** | | | **Scan operand for "1" and combine with RLO through logic OR.**<br>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. **1** |
| **AN** | | | **Scan operand for "0" and combine with RLO through logic AND.**<br>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. **1** |
| **ON** | | | **Scan operand for "0" and combine with RLO through logic OR.**<br>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. **1** |

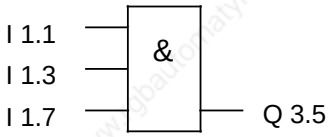
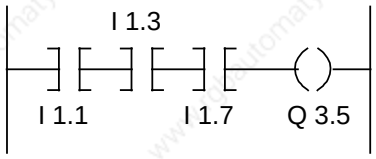
| **ID** | **Parameter** | **CPU 941/942/943/944** |
|---|---|---|
| I | | 0.0 to 127.7 |
| Q | | 0.0 to 127.7 |
| F | | 0.0 to 255.7 |
| T | | 0 to 127 |
| C | | 0 to 127 |

1 If the scan follows an RLO limiting operation directly (first scan), the scan result is reloaded as a new RLO.

### AND Operation

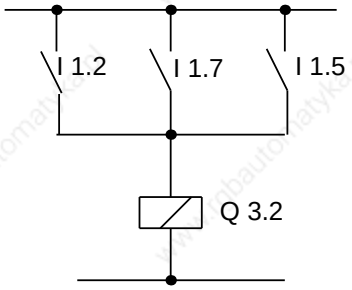
The AND operation scans to see if various conditions are satisfied simultaneously.

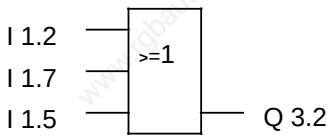
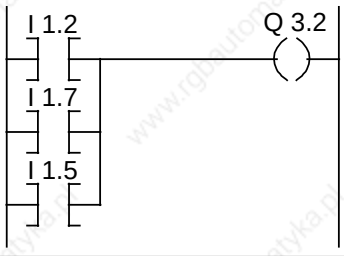
| Example | Circuit Diagram |
|---|---|
| Output Q 3.5 is "1" when all three inputs are "1".<br>The output is "0" if at least one input is "0".<br>The number of scans and the sequence of the logic statements are optional. |  |

| STL | CSF | LAD |
|---|---|---|
| A I 1.1<br>A I 1.3<br>A I 1.7<br>= Q 3.5 |  |  |

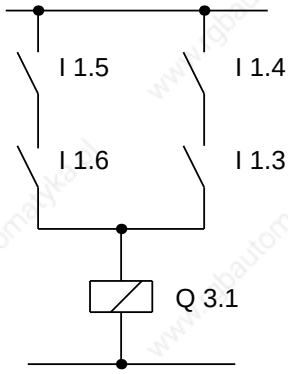
### OR Operation

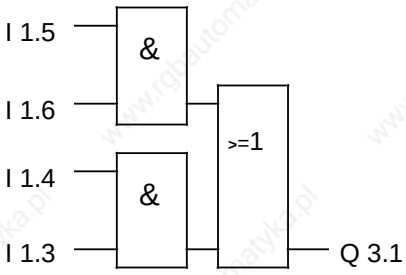
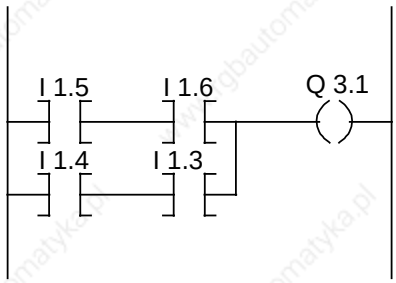
The OR operation scans to see if one of two (or more) conditions has been satisfied.

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

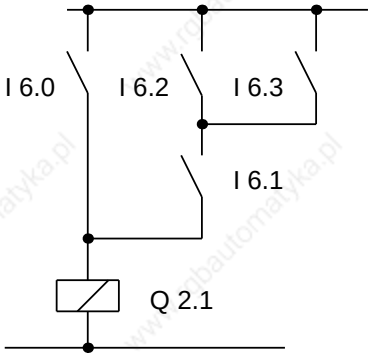
| STL | CSF | LAD |
|---|---|---|
| O I 1.2<br>O I 1.7<br>O I 1.5<br>= Q 3.2 |  |  |

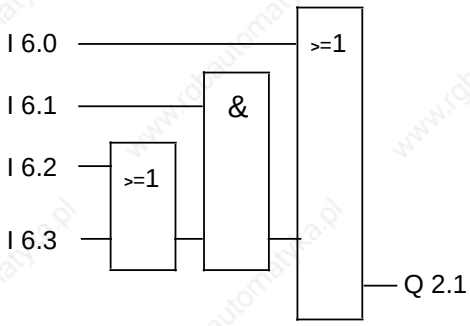
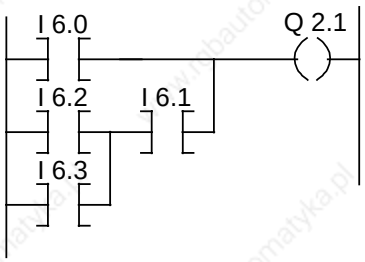
**AND before OR Operation**

| Example | Circuit Diagram |
|---|---|
| Output Q 3.1 is "1" when at least one AND condition has been satisfied.<br>Output Q 3.1 is "0" when neither of the two AND conditions has been satisfied. |  |

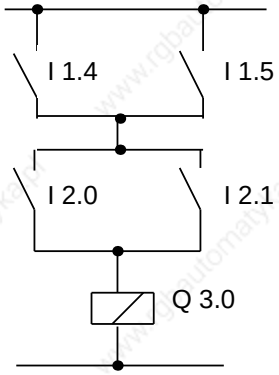
| STL | CSF | LAD |
|---|---|---|
| A I 1.5<br>A I 1.6<br>O<br>A I 1.4<br>A I 1.3<br>= Q 3.1 |  |  |

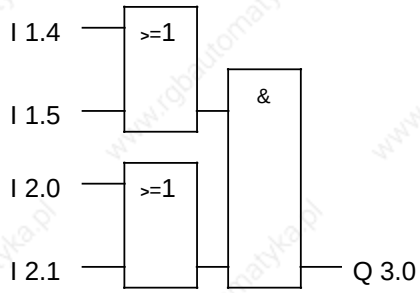
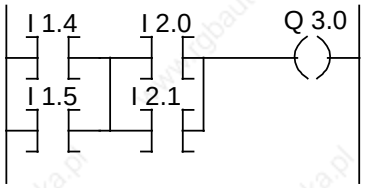
**OR before AND Operation**

<table>
<tr><th>Example</th><th>Circuit Diagram</th></tr>
<tr><td>Output Q 2.1 is "1" when one of the following conditions has been satisfied:<br>• input I 6.0 is "1".<br>• input I 6.1 and either input I 6.2 or I 6.3 are "1".<br>Output Q 2.1 is "0" when none of the AND conditions has been satisfied.</td><td> 
</td></tr>
</table>

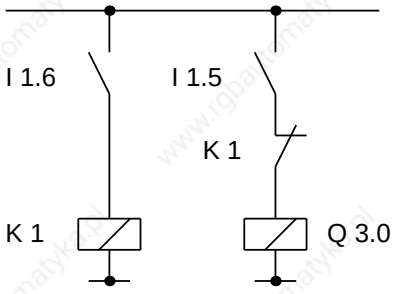
| STL | CSF | LAD |
|---|---|---|
| O I 6.0<br>O<br>A I 6.1<br>A(<br>O I 6.2<br>O I 6.3<br>)<br>= Q 2.1 |  |  |

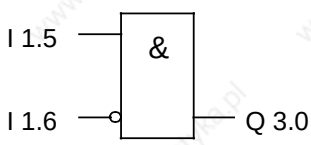
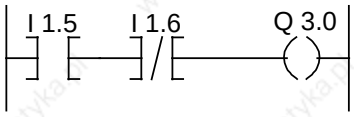
**OR before AND Operation**

| Example | Circuit Diagram |
|---|---|
| Output Q 3.0 is "1" when both OR conditions have been satisfied.<br>Output Q 3.0 is "0" when at least one OR condition has not been satisfied. |  |

| STL | CSF | LAD |
|---|---|---|
| A(<br>O I 1.4<br>O I 1.5<br>)<br>A(<br>O I 2.0<br>O I 2.1<br>)<br>= 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 closed contact activated) and input I 1.6 is "0" (normally closed contact not activated). |  |

| STL | CSF | LAD |
|---|---|---|
| A I 1.5<br>AN I 1.6<br>= Q 3.0 |  |  |

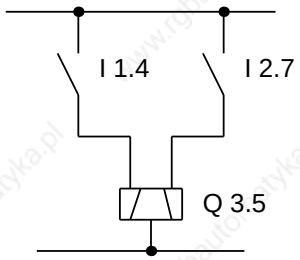
### 8.1.2 Set/Reset Operations

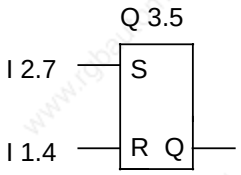
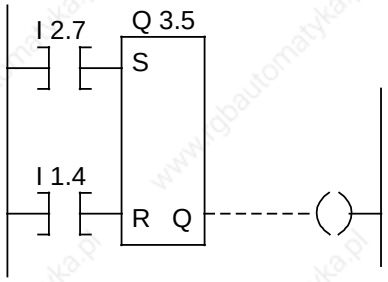
Set/reset operations store the result of logic operation (RLO) formed in the processor. The stored RLO represents the signal state of the addressed operand. Storage can be dynamic (assignment) or static (set and reset). Table 8-2 provides an overview of the set/reset operations. Examples follow the table.

**Table 8-2. Overview of the Set/Reset Operations**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **S** | | | **Set**<br>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**<br>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**<br>Every time the program is scanned, the current RLO is assigned to the addressed operand. |
| **ID** | | **Parameter** | **CPU 941/942/943/944** |
| | I | | 0.0 to 127.7 |
| | Q | | 0.0 to 127.7 |
| | F | | 0.0 to 255.7 |

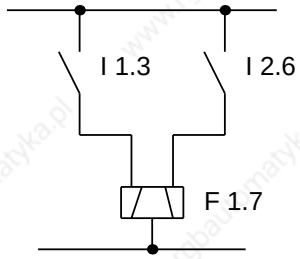
**Flip-Flop for a Latching Signal Output**

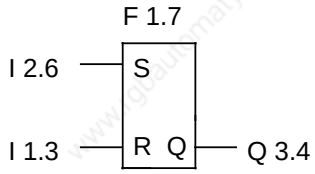
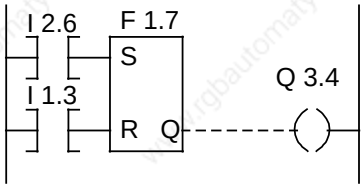
| Example | Circuit Diagram |
|---|---|
| A "1" at input I 2.7 sets flip-flop Q 3.5 (signal state "1"). If the signal state at input I 2.7 changes to "0", the state of output Q 3.5 is maintained, i.e., the signal is latched.<br>A "1" at input I 1.4 resets the flip-flop (signal state "0"). When the "SET" 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.<br>In this example, resetting output Q 3.5 has priority. |  |

| STL | CSF | LAD |
|---|---|---|
| Q I 2.7<br>S Q 3.5<br>Q I 1.4<br>R Q 3.5<br>**NOP 0 *** |  |  |

* **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 675U, PG 685, or PG 695 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 1.7 (signal state "1").<br>If the signal state at input I 2.6 changes to "0", the state of flag F 1.7 is maintained, i.e., the signal is latched.<br>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 1.7 maintains signal state "0".<br>The signal state of the flag is scanned and transferred to output Q 3.4. |  |

| STL | CSF | LAD |
|---|---|---|
| A I 2.6<br>S F 1.7<br>A I 1.3<br>R F 1.7<br>A F 1.7<br>= Q 3.4 |  |  |

### 8.1.3 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 S5-115U they are each 16 bits long. The accumulators are structured as shown in Figure 8-1.

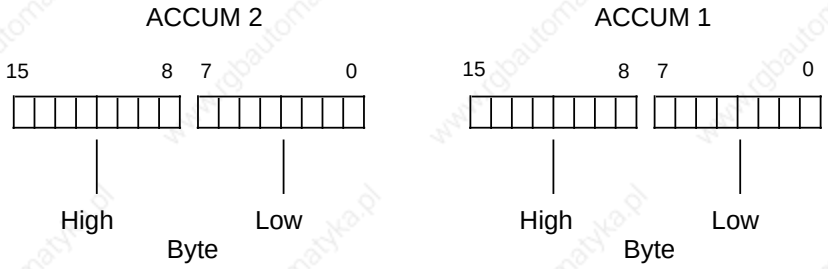



**Figure 8-1. Accumulator Structure**

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 8-3 provides an overview of the load and transfer operations. Examples follow the table.

**Table 8-3. Overview of Load and Transfer Operations**

| Operation | Operand | Meaning |
|---|---|---|
| **L** | | **Load**<br>The operand contents are copied into ACCUM 1 regardless of the RLO.<br>The RLO is not affected. |
| **T** | | **Transfer**<br>The contents of ACCUM 1 are assigned to an operand regardless of the RLO.<br>The RLO is not affected. |
| **ID** | **Parameter** | **CPU 941/942/943/944** |
| IB | | 0 to 127 |
| IW | | 0 to 126 |
| QB | | 0 to 127 |
| QW | | 0 to 126 |
| FY | | 0 to 255 |
| FW | | 0 to 254 |
| DR | | 0 to 255 |
| DL | | 0 to 255 |
| DW | | 0 to 255 |
| T[1] | | 0 to 127 |
| C[1] | | 0 to 127 |
| PB | | 0 to 127<br>128 to 255 |
| PW | | 0 to 126<br>128 to 254 |
| KM[1] | | random bit pattern (16 bits) |
| KH[1] | | 0 to FFFF |
| KF[1] | | - 32768 to +32767 |
| KY[1] | | 0 to 255 per byte |
| KB[1] | | 0 to 255 |
| KS[1] | | any 2 alphanumeric characters |
| KT[1] | | 0.0 to 999.3 |
| KC[1] | | 0 to 999 |
| **LD** | | **Load in BCD**<br>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 These operands cannot be used for transfer

**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 (see Figure 8-2).

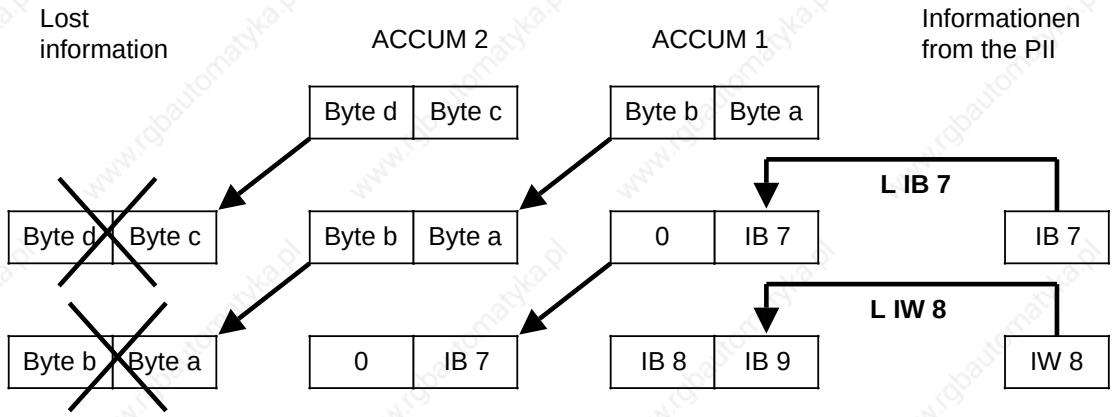



**Figure 8-2. Execution of the Load Operation**

**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 8-3 shows how byte a, the low byte in ACCUM 1, is transferred to QB 5.

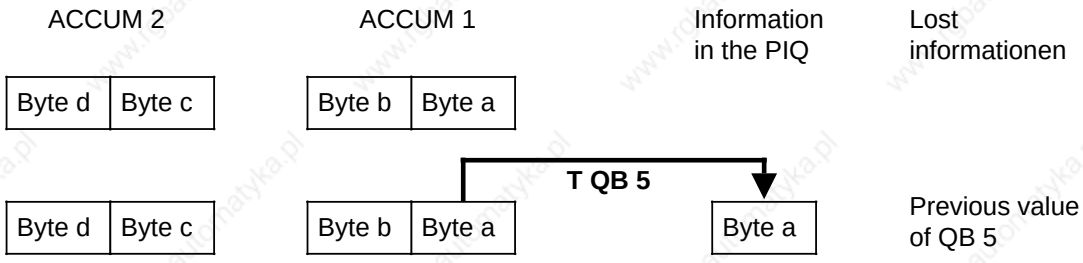
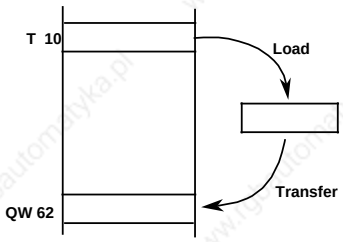
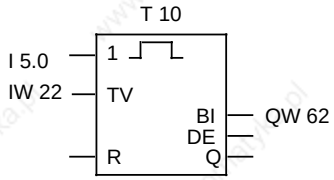
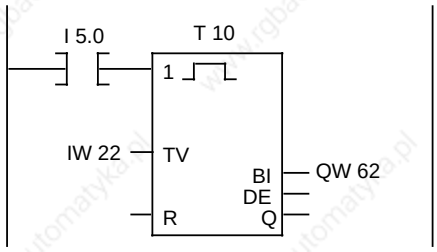


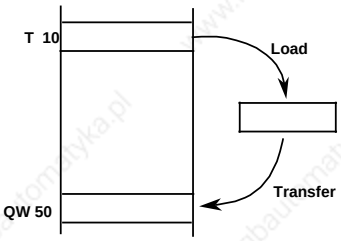
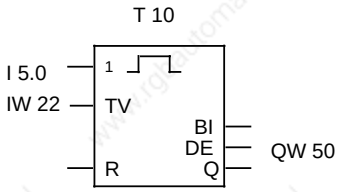
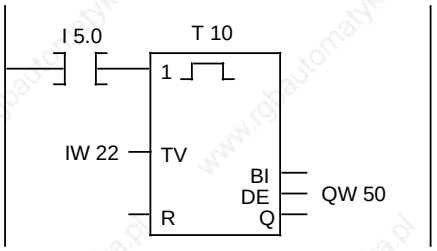

**Figure 8-3. Transferring a Byte**

**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.<br>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.<br>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<br>L IW 22<br>SP T 10<br>**NOP 0**<br>L T 10<br>T QW 62<br>**NOP 0**<br>**NOP 0** |  |  |

**Loading and Transferring a Time (Coded)**

<table>
<tr><th colspan="3">Example</th><th>Representation</th></tr>
<tr><td colspan="3">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.</td><td> 
</td></tr>
<tr><th>STL</th><th>CSF</th><th colspan="2">LAD</th></tr>
<tr><td>A I 5.0<br>L IW 22<br>SP T 10<br>NOP 0<br>NOP 0<br>LC T 10<br>T QW 50<br>NOP 0</td><td> 
</td><td colspan="2"> 
</td></tr>
</table>

### 8.1.4 Timer Operations

The program uses timer operations to implement and monitor chronological sequences. Table 8-4 provides an overview of timer operations. Examples follow the table.

**Table 8-4. Overview of Timer Operations**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **SP** | | | **Pulse Timer**<br>The timer is started on the leading edge of the RLO.<br>When the RLO is "0", the timer is set to "0".<br>Scans result in signal state "1" as long as the timer is running. |
| **SE** | | | **Extended Pulse Timer**<br>The timer is started on the leading edge of the RLO.<br>When the RLO is "0", the timer is not affected.<br>Scans result in signal state "1" as long as the timer is running. |
| **SR** | | | **On-Delay Timer**<br>The timer is started on the leading edge of the RLO.<br>When the RLO is "0", the timer is set to "0".<br>Scans result in signal state "1" when the timer has run out and the RLO is still pending at the input. |
| **SS** | | | **Stored On-Delay Timer**<br>The timer is started on the leading edge of the RLO.<br>When the RLO is "0", the timer is not affected.<br>Scans result in signal state "1" when the timer has run out.<br>The signal state becomes "0" when the timer is reset with the "R" operation. |
| **SF** | | | **Off-Delay Timer**<br>The timer is started on the trailing edge of the RLO.<br>When the RLO is "1", the timer is set to its initial value.<br>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**<br>The timer is reset to its initial value as long as the RLO is "1".<br>When the RLO is "0", the timer is not affected.<br>Scans result in signal state "0" as long as the timer is reset or has not been started yet. |
| **ID** | T | | **Parameter**<br>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 s | 0.1 s | 1 s | 10 s |

**Example:** KT 40.2 corresponds to 40 x 1 sec.

Tolerance:

The time tolerance is equivalent to the time base.

| 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. to 40 sec. |
| | KT 40.2 | 40 x 1 sec. - 1 sec. 39 sec. to 40 sec. |
| | KT 4.3 | 4 x 10 sec. - 10 sec. 30 sec. to 40 sec. |

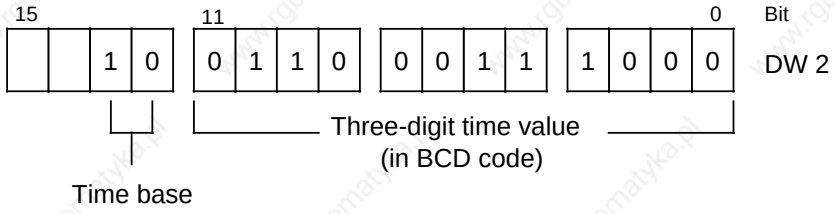
**Note**

Always use the smallest time base possible.

**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 | 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 DB3.

```
C   DB3
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 (see Figure 8-4).
Use the "Load in BCD" operation for digital display output.

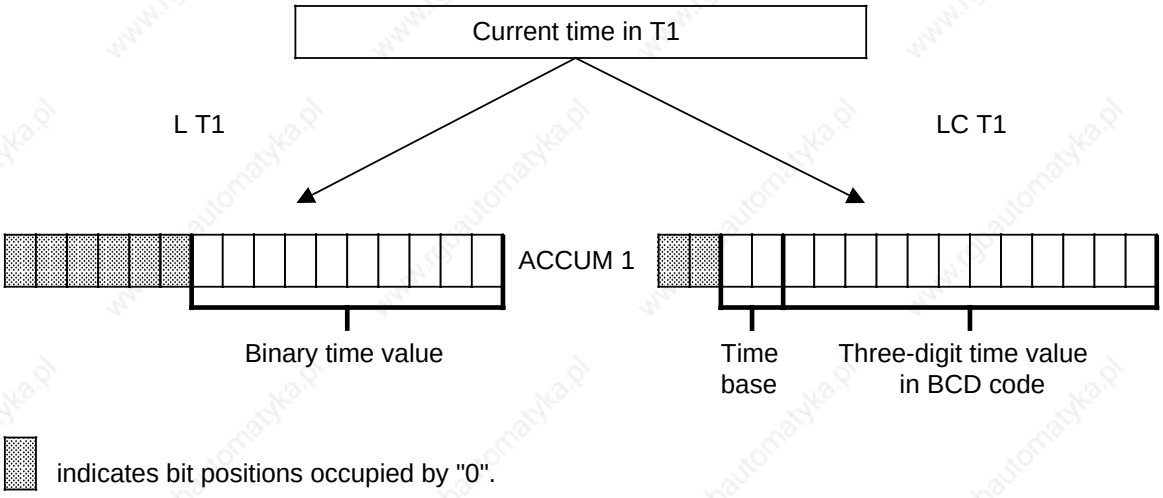


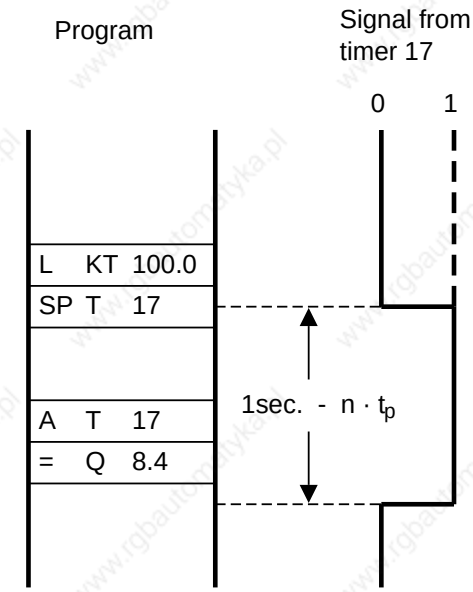

**Figure 8-4. Output of the Current Time (Example)**

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 |
|---|---|
|  <br>n: number of program scanning cycles<br>$t_p$: 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.<br><br>* KT 100.0 is equal to 1 sec. |

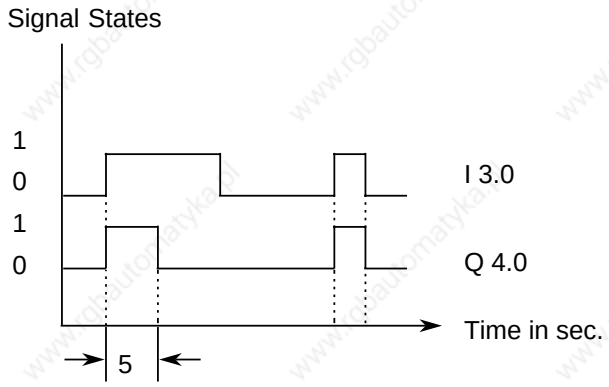
Except for "Reset timer," all timer operations are started only on an edge of the RLO. (The RLO alternates between "0" and "1".)
After being started, the loaded time is decremented in units corresponding to the time base until it reaches zero.
If there is an edge change while the timer is running, the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

**Pulse**

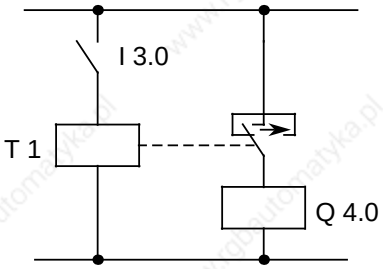
**Example:**

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**

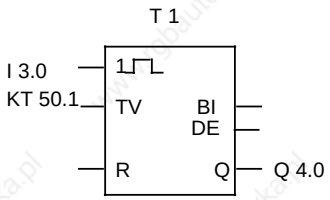



T 1: Time relay with transitional NO contact

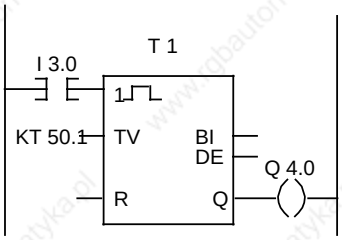
**STL**

```
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**




**Note**

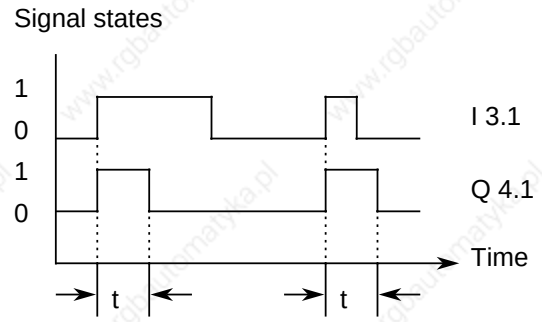
The time values are unsharp at the height of the timebase. The retentive ON delay can be retriggered.

**Extended pulse**

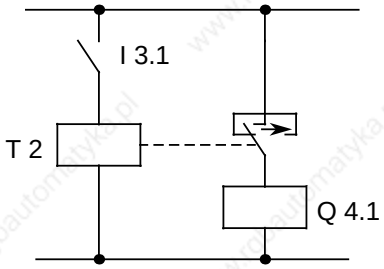
**Example:**

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**

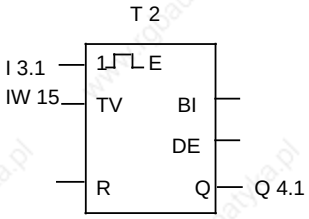



T 2: Time relay with pulse shaper

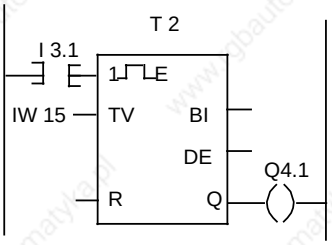
**STL**

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

**CSF**




**LAD**

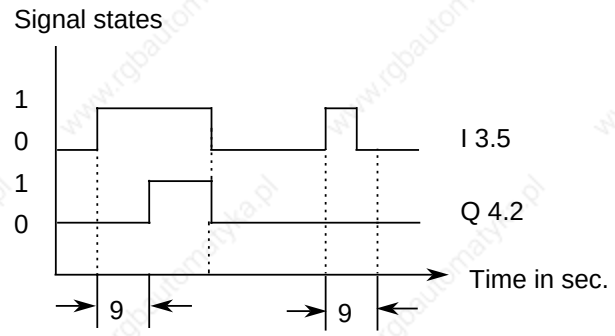


**On-delay**

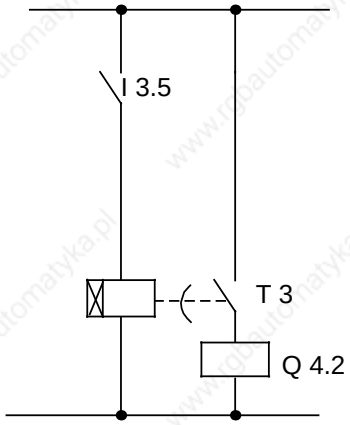
**Example:**

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**



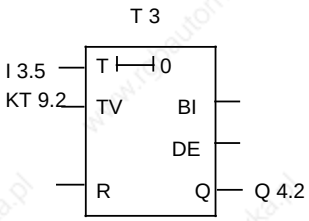

**Circuit Diagram**



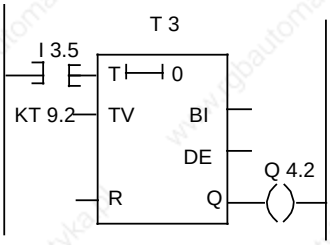

**STL**

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

**CSF**




**LAD**




**Note**

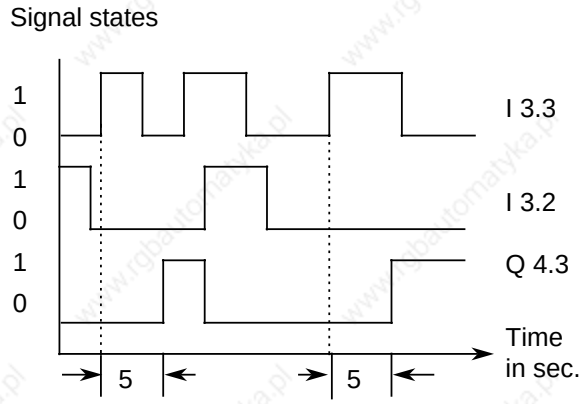
The time value "9 sec." will have a sharper tolerance if you load the timer with the statement "L KT 900.0".

**Stored On-Delay and Reset**

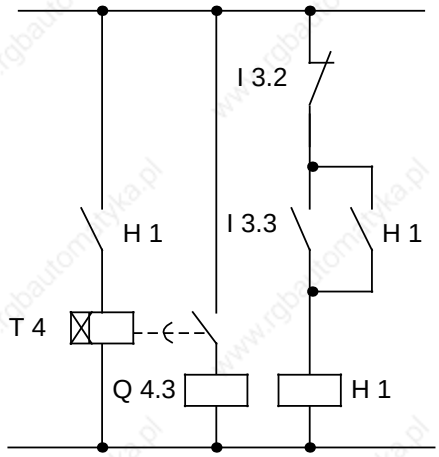
**Example:**

Output Q 4.3 is set 5 sec. after 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**

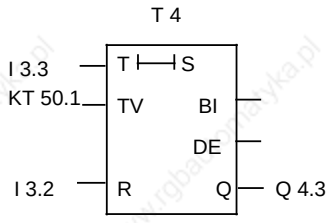



H 1: Auxiliary relay

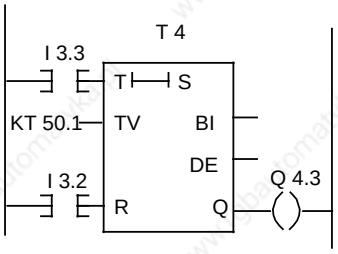
**STL**

```
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**




## Note

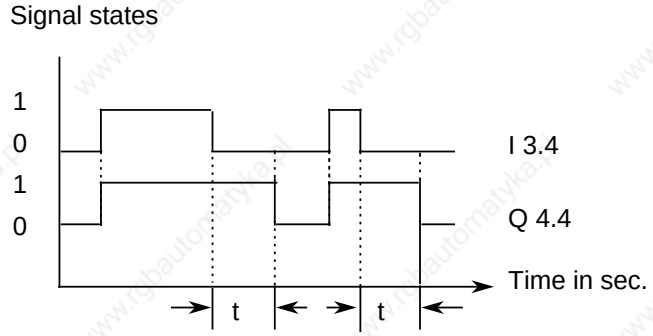
The time values are unsharp at the height of the timebase. The retentive ON delay can be retriggered.

**Off-Delay**

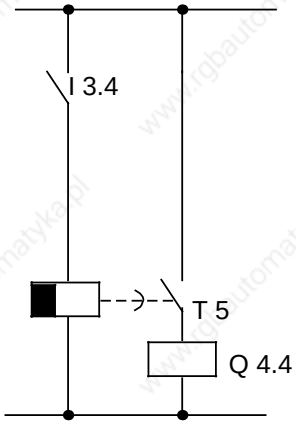
**Example:**

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**



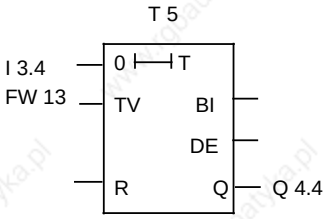

**Circuit Diagram**



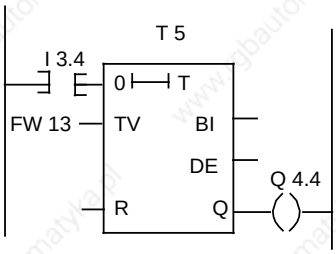

**STL**

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

**CSF**




**LAD**



## 8.1.5 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 8-5 provides an overview of the counter operations. Examples follow the table.

**Table 8-5. Overview of Counter Operations**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **S** | | | **Set Counter**<br>The counter is set on the leading edge of the RLO. |
| **R** | | | **Reset Counter**<br>The counter is set to zero as long as the RLO is "1". |
| **CU** | | | **Count Up**<br>The count is incremented by 1 on the leading edge of the RLO.<br>When the RLO is "0", the count is not affected. |
| **CD** | | | **Count Down**<br>The count is decremented by 1 on the leading edge of the RLO.<br>When the RLO is "0", the count is not affected. |
| **ID**<br>C | | **Parameter**<br>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. | |

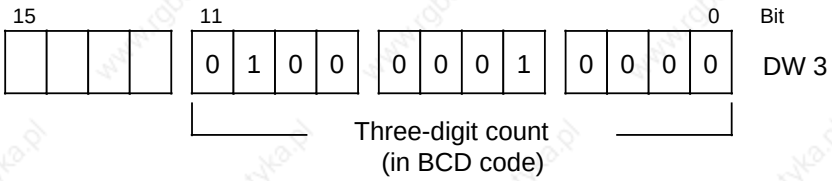
### Loading a Constant Count

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 (see Figure 8-5).
The "Load in BCD" operation is suitable for output via a numeric display.

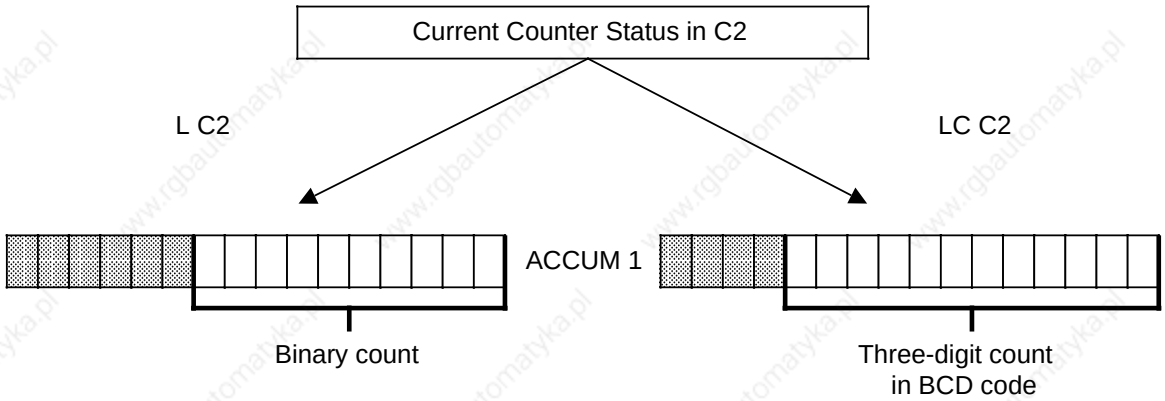


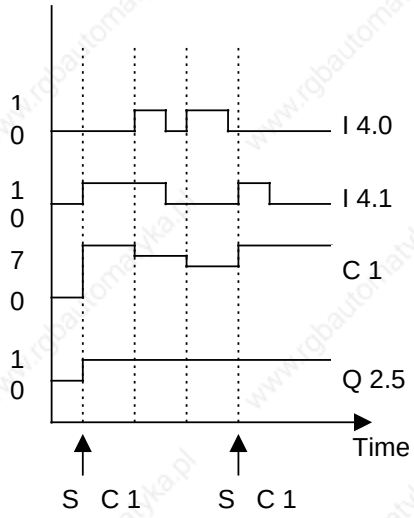

**Figure 8-5. Outputting the Current Counter Status**

**Setting a Counter "S" and Counting Down "CD"**

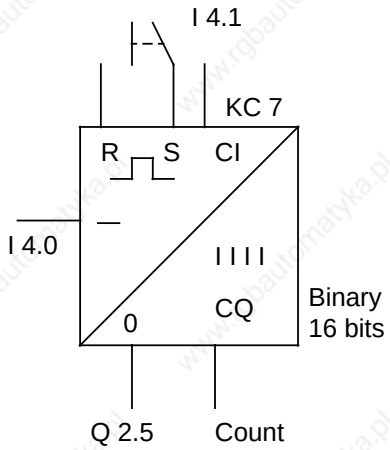
**Example:**

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**



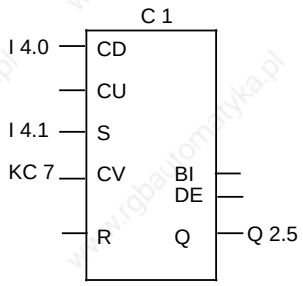

**Circuit Diagram**



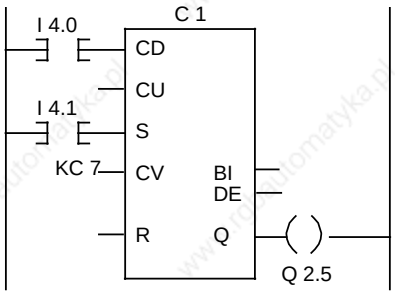

**STL**

```
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**

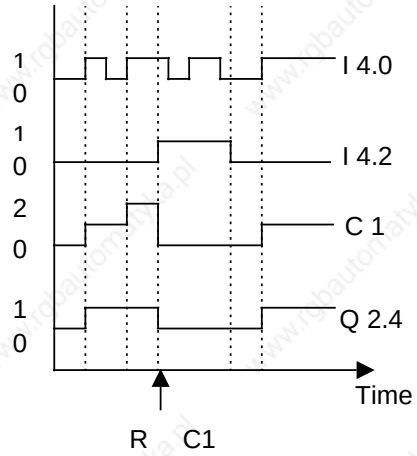


**Resetting a Counter "R" and Counting Up "CU"**

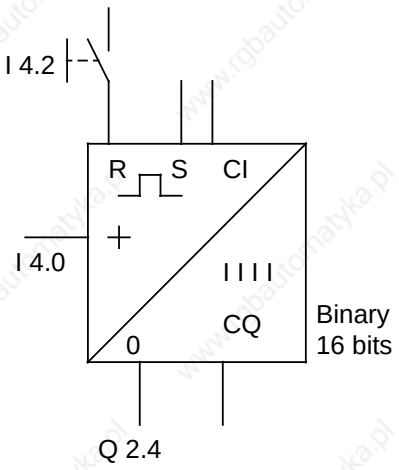
**Example:**

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**



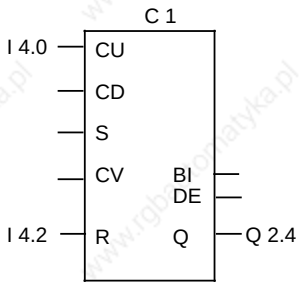

**Circuit Diagram**



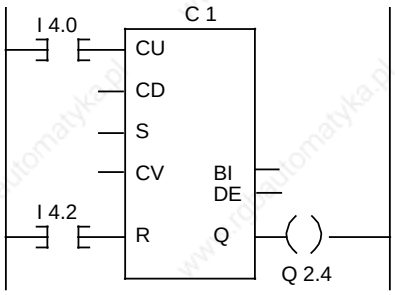

**STL**

```
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**



## 8.1.6 Comparison Operations

Comparison operations compare the contents of the two accumulators. The comparison does not change the accumulators' contents. Table 8-6 provides an overview of the comparison operations. An example follows the table.

**Table 8-6. Overview of Comparison Operations**

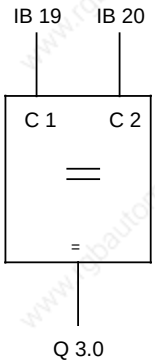
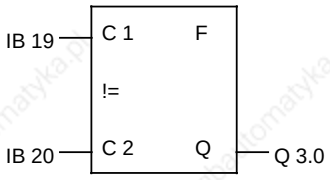
| Operation | Operand | | Meaning |
|---|---|---|---|
| **! = F** | | | **Compare for "equal to"**<br>The contents of the two accumulators are interpreted as bit patterns and scanned to see if they are equal. |
| **> < F** | | | **Compare for "not equal to"**<br>The contents of the two accumulators are interpreted as bit patterns and compared to see if they are not equal. |
| **> F** | | | **Compare for "greater than"**<br>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"**<br>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"**<br>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"**<br>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 (see Section 8.4).

> **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<br>L IB 20<br>!=F<br>= Q 3.0 |  |

## 8.1.7 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 8-7 provides an overview of the arithmetic operations. An example follows the table. The S5-115U has integrated function blocks for multiplication and division.

**Table 8-7. Overview of Arithmetic Operations**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **+ F** | | | **Addition**<br>The contents of both accumulators are added. |
| **- F** | | | **Subtraction**<br>The contents of ACCUM 1 are subtracted from the contents of ACCUM 2. |

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

**Processing an Arithmetic Operation**

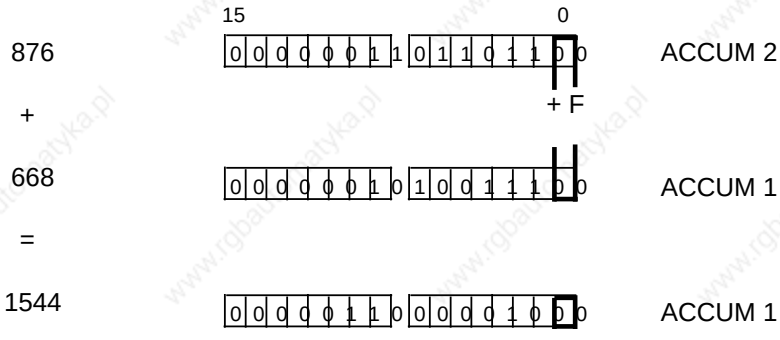
Before an arithmetic operation is executed, both operands must be loaded into the accumulators.

> **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**




## 8.1.8 Block Call Operations

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

**Table 8-8. Overview of Block Call Operations**

| Operation | Operand | Meaning |
|---|---|---|
| **JU** | | **Jump unconditionally**<br>Program scanning continues in a different block regardless of the RLO.<br>The RLO is not affected. |
| **JC** | | **Jump conditionally**<br>Program scanning jumps to a different block when the RLO is "1". Otherwise program scanning continues in the previous block.<br>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**<br>A data block is activated regardless of the RLO.<br>Program scanning is not interrupted.<br>The RLO is not affected. |
| **G** | | **Generate and delete a data block ***<br>An area is set up in the RAM to store data regardless of the RLO. |
| **ID** | **Parameter** | |
| DB | 2 to 255 ** | |
| **BE** | | **Block end**<br>The current block is terminated regardless of the RLO.<br>Program scanning continues in the block in which the call originated. The RLO is "carried along" but not affected. |
| **BEU** | | **Block end, unconditional**<br>The current block is terminated regardless of the RLO.<br>Program scanning continues in the block in which the call originated. The RLO is "carried along" but not affected. |
| **BEC** | | **Block end, conditional**<br>When the RLO is "1", the current block is terminated.<br>Program scanning continues in the block in which the call originated. During the block change, the RLO remains "1".<br>If the RLO is "0", the operation is not executed.<br>The RLO is set to "1" and linear program scanning continues. |

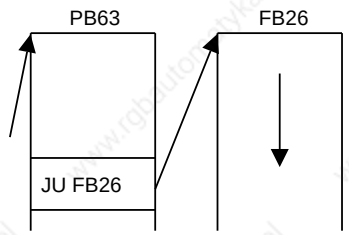
* The length of the DB must be deposited in ACCUM 1 before execution of the command. In the case of length 0, the DB is invalid.

** Data blocks DB0 and DB1 are reserved.

**Unconditional Block Call "JU"**

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

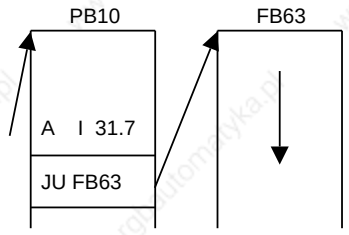
**Example:** A special function has been programmed in FB26. It is called at several locations in the program, e.g., in PB63, and processed.

| Program Sequence | STL | Explanation |
|---|---|---|
|  | .<br>.<br>.<br>.<br>.<br>JU FB26<br>. | The "JU FB26" statement in program block PB63 calls function block FB26. |

**Conditional Block Call "JC"**

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

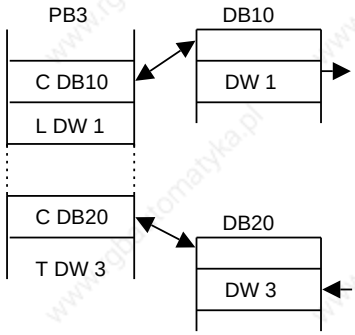
**Example:** A special function has been programmed in FB63. It is called and processed under certain conditions, e.g., in PB10.

| Program Sequence | STL | Explanation |
|---|---|---|
|  | .<br>.<br>.<br>S F 1.0<br>A I 31.7<br>JU FB63<br>. | The "JC FB63" statement in program block PB10 calls function block FB63 if input I 31.7 is "1". |

**Call 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 PB3 needs information that has been programmed as data word DW 1 in data block DB10. Other data, e.g., the result of an arithmetic operation, is stored as data word DW 3 in data block DB20.

| Program Sequence | STL | Explanation |
|---|---|---|
|  | C DB10<br>L DW 1<br>.<br>.<br>.<br>C DB20<br>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 DB20. |

**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 (see the example below).
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 (see 11.1.4 and 11.1.5).

**Note**

The block is retained as a "dead" block until the PLC memory is compressed (see Section 7.5.3).

If an already existing data block is to be set up, the statement G DBx will have no effect! If the DB to be set up is longer than the available memory, the CPU goes to STOP with "TRAF" or jumps to the relevant error response OB.
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<br>**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.<br>Data block 5 is generated with a length of 128 data words (0000) in the RAM of the PC 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. |

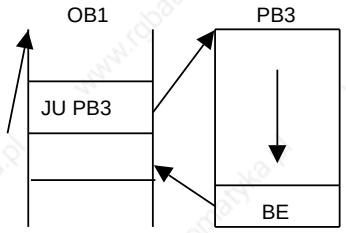
**Deleting a Data Block**

| Example | STL | Explanation |
|---|---|---|
| Delete a data block that is no longer needed. | L KF + 0<br>**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.<br>Data block 5, which is in the RAM of the PC, 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.

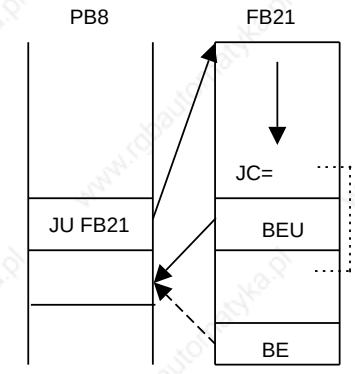
**Example:** Program block PB3 is terminated by the "BE" statement.

| Program Sequence | STL | Explanation |
|---|---|---|
|  | .<br>.<br>.<br>.<br>.<br>.<br>BE | The "BE" statement terminates program block PB 3 and causes program scanning to return to organization block OB1. |

**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 (see Sections 8.2.10 and 8.3.4).
Binary logic operations cannot be continued in a higher-order block.

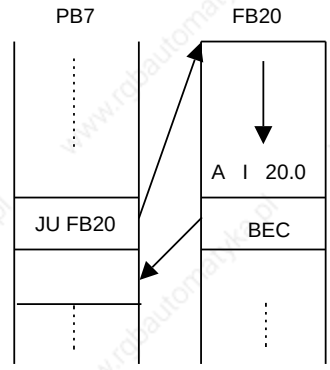
**Example:** Scanning of function block FB21 is terminated regardless of the RLO.

| Program Sequence | STL | Explanation |
|---|---|---|
|  | .<br>.<br>.<br>.<br>JC=<br>BEU<br>.<br>.<br>.<br>.<br>BE | The "BEU" statement causes program scanning to leave function block FB21 and return to program block PB8. |

**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 |
|---|---|---|
|  | .<br>.<br>.<br>.<br>S Q 1.0<br>A I 20.0<br>BEC<br>.<br>.<br>.<br>. | The "BEC" statement causes program scanning to return to program block PB7 from function block FB20 if input I 20.0 is "1." |

## 8.1.9 Other Operations

Table 8-9 lists other basic operations. Explanations follow the table.

**Table 8-9. Other Operations**

| Operation | Operand | Meaning |
|---|---|---|
| **STP** | | **Stop at the end of program scanning.**<br>Current program scanning is terminated.<br>The PIQ is read out. Then the PC goes into the "STOP" mode. |
| **NOP 0** | | **"No" Operation**<br>Sixteen bits in the RAM are set to "0". |
| **NOP 1** | | **"No" Operation**<br>Sixteen bits in the RAM are set to "1". |
| **BLD** | ↑ ↑ | **Display Generation Operation**<br>"BLD" means a display generation operation for the programmer. |
| **ID** | | **Parameter**<br>130, 131, 132, 133, 255 |

**Note**

These operations can be programmed in STL form only.

**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.

**Display Generation Operations**

"BLD" display generation operations divide program parts into segments within a block.

"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.

## 8.2 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 8.2.1 through 8.2.11 describe the supplementary operations.

### 8.2.1 Load Operation

As with the basic load operations, the supplementary load operation copies information into the accumulator. Table 8-10 explains the load operation. An example follows the table.

**Table 8-10. Load Operation**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **L** | | | **Load**<br>A word from the system data is loaded into ACCUM 1 regardless of the RLO. |
| **ID** | | **Parameter** | |
| | RS | 0 to 255 | |

| Example | STL | Explanation |
|---|---|---|
| In the event of a time-out, the error address is stored in SD 103. An important output module is plugged in at start address 4. If the time-out is triggered by this address, the CPU is to go into the "STOP" mode. Otherwise a signal is to be given and program scanning is to continue. You can program this example in OB24. | L RS 103<br>L KH F004<br><br><br><> F<br>= Q 12.0<br>BEC<br><br><br>STP | The contents of SD 103 and the address of the important module are loaded into the accumulators.<br>If the two values are not equal, output Q 12.0 is set.<br>Program scanning continues in OB1 (or in the block where the call originated).<br>If the two values are equal, the CPU goes into the "STOP" mode. |

## 8.2.2 Enable Operation

Use the enable operation (FR) to execute the following operation even without edge change:
- start a timer
- set a counter
- count up and down.

Table 8-11 presents the enable operation. An example follows the table.

**Table 8-11. Enable Operation**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **FR** | | | **Enable a Timer/Counter**<br>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<br>L<br>SE<br>A<br>= | I<br>KT<br>T<br>T<br>Q | 2.5<br>5.3 *<br>2<br>2<br>4.2 | Start a timer T 2 as extended pulse.<br>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<br>**FR**<br><br>BE | Q<br>**T**<br><br> | 3.4<br>**2**<br><br> | 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.<br>If input I 2.5 is not set during the edge change of output Q 3.4, the timer is not restarted. |

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

### 8.2.3 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 8-12 provides an overview of these operations.

**Table 8-12. Overview of Bit Operations**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **TB** | | | **Test a bit for signal state "1"**<br>A single bit is scanned regardless of the RLO. The RLO is affected according to the bit's signal state (see Table 8-13). |
| **TBN** | | | **Test a bit for signal state "0"**<br>A single bit is scanned regardless of the RLO. The RLO is affected according to the bit's signal state (see Table 8-13). |
| **SU** | | | **Set a bit unconditionally**<br>The addressed bit is set to "1" regardless of the RLO. The RLO is not affected. |
| **RU** | | | **Reset a bit unconditionally**<br>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 8-13 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 8-13. Effect of "TB" and "TBN" on the RLO**

| Operation | P | | PN | |
|---|---|---|---|---|
| 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 FB5 or FB6. After 800 pieces, counter 10 is to be reset automatically and start counting again. | C DB 10<br>A I 2.0<br>CU C 10<br>A I 3.0<br>L KC 0<br>S C 10<br>O I 4.0<br>O F 5.2<br>R C 10<br>LC C 10<br>T DW 12<br><br>**TBN D 12.8**<br><br>JC FB 5<br><br><br>**TB D 12.8**<br><br>JC FB 6<br><br><br>**TB D 12.11**<br><br>= F 5.2 | Call data block 10.<br><br>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.<br><br>As long as bit 8 of data word DW 12 is zero, program processing jumps to function block FB5. This is the case for the first, third, fifth etc. batch of 100 pieces.<br>As long as bit 8 of data word DW 12 is "1", program scanning jumps to function block FB6. This is the case for the second, fourth, sixth etc. batch of 100 pieces.<br>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<br>:CU C 20<br>:A I 11.0<br>:L KC 0<br>:S C 20<br><br>**:TB C 20.8**<br><br>:JC = FULL<br>:BEU<br><br>**FULL:RU C 20.8**<br>: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.<br><br>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.

### 8.2.4 Digital Logic Operations

Digital logic operations combine the contents of both accumulators logically bit by bit.
Table 8-14 provides an overview of these digital logic operations. Examples follow the table.

**Table 8-14. 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**

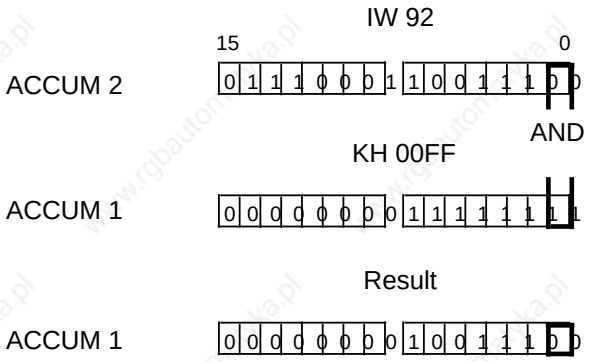
A digital logic operation is executed regardless of the RLO. It also does not affect the RLO. However, it sets condition codes according to the result of the arithmetic operation (see Section 8.4).

> **Note**
>
> Make sure 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. |

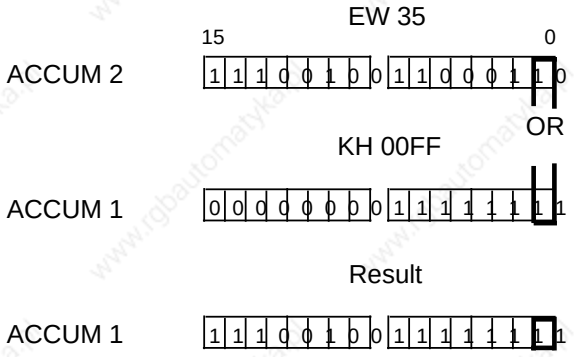
**Numeric Example**




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. |

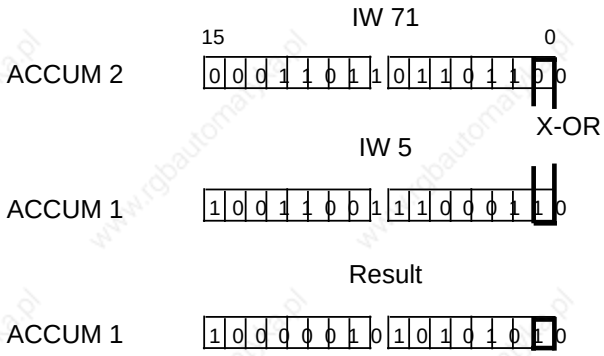
**Numeric Example**




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.

## 8.2.5 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 8-15 provides an overview of the shift operations. Examples follow the table.

**Table 8-15. Overview of Shift Operations**

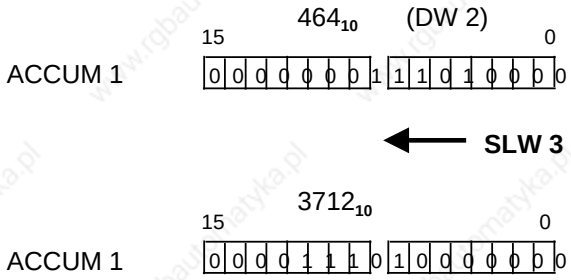
| Operation | Operand | | Meaning |
|---|---|---|---|
| **SLW** | | | **Shift to the left.**<br>The bit pattern in ACCUM 1 is shifted to the left. |
| **SRW** | | | **Shift to the right.**<br>The bit pattern in ACCUM 1 is shifted to the right. |
| | | **Parameter** | 0 to 15 |

**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. Following execution of the operation, the state of bit $2^0$ (SRW) or bit $2^{15}$ (SLW) has an influence on the CC1 bit, which can then be evaluated.
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. |

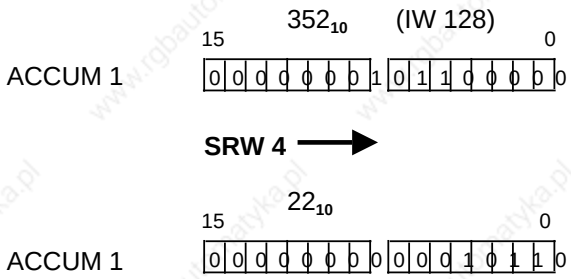
**Numeric Example**




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.

## 8.2.6 Conversion Operations

Conversion operations convert the values in ACCUM 1. Table 8-16 provides an overview of the conversion operations. Examples follow the table.

**Table 8-16. Overview of Conversion Operations**

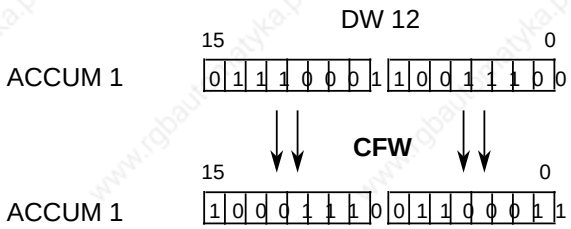
| Operation | Operand | Meaning |
|---|---|---|
| **CFW** | | **One's complement**<br>The contents of ACCUM 1 are inverted bit by bit. |
| **CSW** | | **Two's complement**<br>The contents of ACCUM 1 are inverted bit by bit. Afterwards the word $0001_H$ is added. |

**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 (see Section 8.4).

| 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**.** |

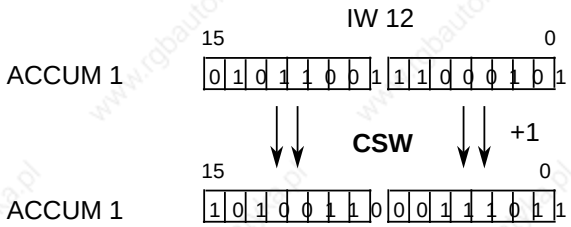
**Numeric Example**




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.

## 8.2.7 Decrement/Increment

The decrement/increment operations change the data loaded into ACCUM 1. Table 8-17 provides an overview of the decrement/increment operations. An example follows the table.

**Table 8-17. Decrement/Increment Operations**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **D** | | | **Decrement**<br>Decrement the contents of the accumulator. |
| **I** | | | **Increment**<br>Increment the contents of the accumulator.<br>The contents of ACCUM 1 are either decremented or incremented by the number indicated in the parameter.<br>Execution of the operation is unconditional and is limited to the right-hand byte (without carry). |
| | | | **Parameter**<br>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 DB6. |
| | 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. |
| | D | 33 | | The result would be $FFF_H$. However, 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 ($10FF_H$). |

## 8.2.8 Disable/Enable Interrupt

The disable/enable interrupt operations affect interrupt and time-controlled program scanning. They prevent process or time interrupts from interfering with the processing of a sequence of statements or blocks. Table 8-18 lists the disable/enable interrupt operations. An example follows the table.

**Table 8-18. Disable/Enable Interrupt Operations**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **IA** | | | **Disable interrupt** |
| **RA** | | | **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".

| Example | STL | | Explanation |
|---|---|---|---|
| Disable interrupt processing in a specific program section and then enable it again. | | . | |
| | | . | |
| | | . | |
| | | . | |
| | | . | |
| | = | Q 7.5 | |
| | **IA** | | Disable interrupt. |
| | A | I2.3 | |
| | | . | |
| | | . | |
| | | . | |
| | 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.

## 8.2.9 Processing Operation

The processing operation (DO) can handle STEP 5 statements in "indexed" form. Use it to change the parameter of an operand while the control program is being scanned. Table 8-19 and the example that follows explain the processing operation.

**Table 8-19. Processing Operations**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **DO** | | | **Process a flag or data word.** |
| **ID** | FW<br>DW | **Parameter**<br>0 to 254<br>0 to 255 | |

**Processing**

The statement "Process flag or data word x" is a two-word statement that is executed independently of the RLO.

It consists of the following two related statements:

- The first statement contains the processing operation and specifies a flag or data word.
- The second statement specifies the operation and the operand ID that the control program is to process. Enter 0 or 0.0 here as parameter.

> **Note**
>
> If a value other than 0 or 0.0 is specified in the CPU 944, this value is replaced by 0 or 0.0.

The control program works with the parameter that is stored in the flag or data word called by the first statement. If you are indexing binary operations, inputs, outputs, or flags, indicate the bit address in the high byte of this word and the byte address in the low byte.
In all other cases, the high byte must be "0".

The following operations can be combined with the processing statement:

| Operations | Explanation |
|---|---|
| A[1], AN, O, ON | Binary logic operations |
| S, R,= | Set/reset operations |
| FR T, RT, SF T, SD T, SI T, SS T, SE T | Timer operations |
| FR C, RC, SC, CD C, CU C | Counter operations |
| L, LC, T | Load and transfer operations |
| JU=, JC=, JZ=, JN=, JP=, JM=, JO= | Jump operations |
| SLW, SRW | Shift operations |
| D, I | Decrement and increment |
| C DB, JU, JU, TNB | Block calls |

1 The "AI" operation becomes the "AQ" operation in combination with "DO DW" or "DO FW" if the byte address in the data or flag word is greater than 127.
CPU 944 is an exception: In this case, outputs are referenced with the operation sequence DO DWX; A Q X.Y or DO FWX; A Q X.Y

Figure 8-6 shows how data word contents specify the parameter of the next statement.

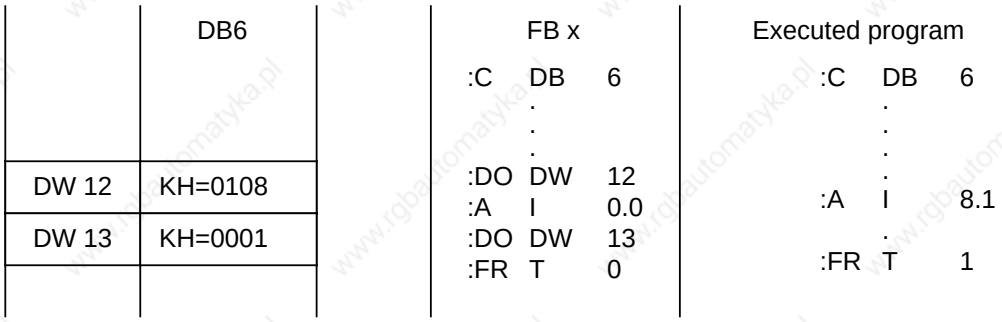



**Figure 8-6. Effect of the Processing Operation**

The following example shows how new parameters are generated each time the program is scanned.

| Example | STL | | | Explanation |
|---|---|---|---|---|
| Set the contents of data words DW 20 to DW 100 to signal state "0". The "index register" for the parameter of the data words is DW 1. | | :C | DB 202 | Call data block 202. |
| | | :L | KB 20 | Load constant 20 into ACCUM 1. |
| | | :T | DW 1 | Transfer contents of ACCUM 1 to DW 1. |
| | L 1 | :L | KH 0 | Load hexadecimal constant 0 into ACCUM 1. |
| | | **:DO** | **DW 1** | Process data word 1. |
| | | **:T** | **DW 0** | Transfer the contents of ACCUM 1 to the data word whose address is stored in data word DW 1. |
| | | :L | DW 1 | Load data word DW 1 into ACCUM 1. |
| | | :L | KB 1 | Load constant 1 into ACCUM 1. Data word DW 1 is shifted to ACCUM 2. |
| | | :+F | | ACCUM 2 and ACCUM 1 are added and the result is stored in ACCUM 1 (incrementing the data word address). |
| | | :T | DW 1 | Transfer the contents of ACCUM 1 to data word DW 1 (new data word address). |
| | | :L | KB 100 | The constant 100 is loaded into ACCUM 1 and the new data word address is shifted to ACCUM 2. |
| | | :<=F | | The accumulators are compared to see if ACCUM 2 ACCUM 1. |
| | | :JC = L 1 | | Jump conditionally to label L1 if ACCUM 2 ACCUM 1. |

## 8.2.10 Jump Operations

Table 8-20 provides an overview of the jump operations. An example follows the table.

**Table 8-20. Overview of Jump Operations**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **JU =** | | | **Jump unconditionally**<br>The unconditional jump is executed independently of conditions. |
| **JC =** | | | **Jump conditionally**<br>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"**<br>The jump is executed only if CC 1=0 and CC 0=0<br>The RLO is not changed. |
| **JN =** | | | **Jump if the result is "not zero"**<br>The jump is executed only if CC 1 CC 0<br>The RLO is not changed. |
| **JP =** | | | **Jump if the result is positive**<br>The jump is executed only if CC 1=1 and CC 0=0<br>The RLO is not changed. |
| **JM =** | | | **Jump if the result is negative**<br>The jump is executed only if CC 1=0 and CC 0=1<br>The RLO is not changed |
| **JO =** | | | **Jump on overflow**<br>The jump is executed if an overflow occurs. Otherwise the jump is not executed. The RLO is not changed. |
| **ID**<br>Jump label (up to 4 characters) | | | |

**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.

| 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.<br>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[1], 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 | |
| | | :L | DW12 | Compare input word IW 1 and data word DW 12. If they are not equal, set RLO to "1". |
| | | :>< | F | |
| | | **:JB=** | **Destination** | If the RLO="1", jump to the "Destination" label. If the RLO="0", process the next statement. |
| | | | . | |
| | | | . | |
| | | | . | |
| | | | . | |
| | DEST. | :A | I 12.2 | |
| | | | . | |
| | | | . | |

1 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.

## 8.2.11 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 (see Section 6.3.4). 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 follow.

**Binary Logic Operations**

Table 8-21 provides an overview of binary logic operations.

**Table 8-21. Overview of Binary Logic Operations**

| Operation | Operand | Meaning |
|---|---|---|
| **A =** | | **AND operation**<br>Scan a formal operand for "1". |
| **AN =** | | **AND operation**<br>Scan a formal operand for "0". |
| **O =** | | **OR operation**<br>Scan a formal operand for "1 ". |
| **ON =** | | **OR operation**<br>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 8-22 provides an overview of the set/reset operations. An example follows the table.

**Table 8-22. Overview of Set/Reset Operations**

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

| Formal operand | Actual Operands Permitted | Parameter Type | Data Type |
|---|---|---|---|
| | Inputs, outputs, and flags addressed in binary form | I , Q | BI |

**Example:** FB30 is assigned parameters in OB1.

| Call in OB1 | | Program in FB30 | | Executed Program | | |
|---|---|---|---|---|---|---|
| | :JU 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 8-23 provides an overview of the load and transfer operations. An example follows the table.

**Table 8-23. 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 flags[1], timers and counters addressed in byte and word form | I , Q<br>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 flags[1] addressed in byte and word form | I, Q | BY, W |

**Example:** FB34 is assigned parameters in PB1.

| Call in PB1 | Program in FB34 | Executed Program |
|---|---|---|
| :JU FB 34<br>NAME :LOAD/TRAN<br>I0 : I 2.0<br>I1 : I 2.1<br>L1 : FW 10<br>LW1 : KC 140<br>LC1 : C 7<br>T1 : QW 4<br>LW2 : KC 160<br>:BE | :A =I0<br>**:L =L1**<br>:S C 6<br>:A =I1<br>**:LW =LW1**<br>:S C 7<br>:A I 2.2<br>:CU C 6<br>:CU C 7<br>**:LC =LC1**<br>**:T =T1**<br>:A I 2.7<br>:R C 6<br>:R C 7<br>**:LW =LW2**<br>**:LC =LC1**<br>:!=F<br>:R C 7<br>:BE | :A I 2.0<br>:L FW 10<br>:S C 6<br>:A I 2.1<br>:L KC 140<br>:S C 7<br>:A I 2.2<br>:CU C 6<br>:CU C 7<br>:LC C 7<br>:T QW 4<br>:A I 2.7<br>:R C 6<br>:R C 7<br>:L KC 160<br>:LC C 7<br>:!=F<br>:R C 7<br>:BE |

1 Data word: DW, DR, DL

### Timer and Counter Operations

Table 8-24 provides an overview of timer and counter operations. Examples follow the table.

**Table 8-24. 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 specified 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<br>NAME:TIME<br>I5 : I 2.5<br>I6 : I 2.6<br>TIM5 : T 5<br>TIM6 : T 6<br>OUT6 : Q 7.6<br>:BE | :AN =I 5<br>:A =I 6<br>:L KT 5.2<br>**:SFD =TIM5**<br>:A =I 5<br>:AN =I 6<br>:L KT 5.2<br>**:SSU =TIM6**<br>:A =TIM5<br>:O =TIM6<br>:= =OUT6<br>:A I 2.7<br>**:RD =TIM5**<br>**:RD =TIM6**<br>**:BE** | :AN I 2.5<br>:A I 2.6<br>:L KT 5.2<br>:SF T 5<br>:A I 2.5<br>:AN I 2.6<br>:L KT 5.2<br>:SS T 6<br>:A T 5<br>:O T 6<br>:= Q 7.6<br>:A I 2.7<br>:R T 5<br>:R T 6<br>:BE |

**Example 2:**

| Function Block Call | Program in Function Block (FB33) | Executed Program |
|---|---|---|
| STL<br>:JU FB 33<br>NAME:COUNT<br>I2 : I 2.2<br>I3 : I 2.3<br>I4 : I 2.4<br>COU5 : C 5<br>OUT3 : Q 7.3<br>:BE | :A =I2<br>:L KC 17<br>**:SEC =COU5**<br>:A =I 3<br>**:SSU =COU5**<br>:A =I 4<br>**:SFD =COU5**<br>:A =COU5<br>:= =OUT3<br>:A I 2.7<br>**:RD =COU5**<br>:BE | :A I 2.2<br>:L KC 17<br>:S C 5<br>:A I 2.3<br>:CU C 5<br>:A I 2.4<br>:CD C 5<br>:A C 5<br>:= Q 7.3<br>:A I 2.7<br>:R C 5<br>:BE |

### Processing Operation

Table 8-25 and the example that follows explain the processing operation.

**Table 8-25. Processing Operation**

| Operation | Operand | Meaning | | |
|---|---|---|---|---|
| **DO =** | | **Process formal operand**<br>The substituted blocks are called unconditionally. | | |
| **Formal operands** | | **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<br>:JU FB 35<br>NAME :PROCES<br>D5 : DB 5<br>DW2 : DW 2<br>D6 : DB 6<br>DW1 : DW 1<br>Q4 : QW 4<br>MOT5 : FB 36<br>:BE | **:DO =D5**<br>:L =DW2<br>**:DO =D6**<br>:T =DW1<br>:T =Q4<br>**:DO =MOT5**<br>:BE | :C DB 5<br>:L DW 2<br>:C DB 6<br>:T DW 1<br>:T QW 4<br>:JU FB 36<br>:BE |

## 8.3 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. Sections 8.3.1 through 8.3.6 describe the system operations.

### 8.3.1 Set Operations

Like the supplementary bit operations, these set operations can change individual bits. Table 8-26 provides an overview of the set operations.

**Table 8-26. Overview of Set Operations**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **SU** | | | **Set bit unconditionally**<br>A specific bit is set to "1" in the system data range. |
| **RU** | | | **Reset bit unconditionally**<br>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.

## 8.3.2 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 8-27 provides an overview of the load and transfer operations.

**Table 8-27. Overview of Load and Transfer Operations**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **LIR** | | | **Load the register indirectly**<br>The contents of a memory word are loaded into the specified register (ACCUM 1, 2). The address is in ACCUM 1. |
| **TIR** | | | **Transfer the register indirectly**<br>The contents of the indicated register are transferred to a memory location. The address is in ACCUM 1. |
| | | | **Parameter**<br>0 (for ACCUM 1), 2 (for ACCUM 2) |
| **LDI** | | | **Load register indirect**<br>The contents of a memory word, whose address is in ACCUM 1, are loaded into the specified accumulator (ACCUM 1, 2) (access to the second memory bank; CPU 944 only). |
| **TDI** | | | **Transfer register indirect**<br>The accumulator specified (ACCUM 1, 2) is transferred to a memory location, whose address is in ACCUM 1, (access to the second memory bank; CPU 944 only). |
| **Code**<br>A1 (for ACCUM 1)<br>A2 (for ACCUM 2) | | | |
| **TNB** | | | **Transfer a data block (byte by byte)**<br>A memory area is transferred in the program memory as a block.<br>End address destination area: ACCUM 1<br>End address source area: ACCUM 2 |
| **T** | | | **Transfer**<br>A word is transferred to the system data range. |
| **ID** | RS | | **Parameter**<br>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 S5-115U 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.

| STL | | | 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. |

**Example:** Loading the contents of memory locations $1231_H$ and $1232_H$ in the second memory bank into ACCUM 2.

Memory location $1231_H$ contains $45_H$;
Memory location $1232_H$ contains $67_H$.

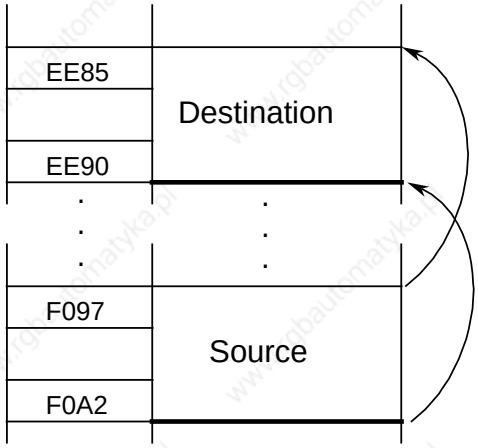
| STL | | | Explanation |
|---|---|---|---|
| L | KH | 1231 | The constant $1231_H$ is loaded into ACCUM 1. |
| LDI | A2 | | ACCUM 2 contains $4567_H$ after this operation, i.e. the contents of memory locations $1231_H$ and $1232_H$. |

**Example:** Transferring the values $44_H$ and $66_H$ to memory locations $1231_H$ and $1232_H$ of the second memory bank.

| STL | | | Explanation |
|---|---|---|---|
| L | KH | 4466 | The constant $4466_H$ is loaded into ACCUM 1. |
| L | KH | 1231 | ACCUM 1 contains $1231_H$ and ACCUM 2 contains $4466_H$ after this operation. |
| TDI | A2 | | Memory location $1231_H$ contains the value $44_H$ and memory location $1232_H$ the value $66_H$ after the transfer operation. |

**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 adress 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. |

**Transferring to the system data area**

**Example:** Set the scan monitoring time to 100 msec. after each mode change from "STOP" to "RUN". Program the time as a multiple of ten in system data word 96*. The following function block can be called from OB21, for example.

| STL | | | Explanation |
|---|---|---|---|
| FB 11 | | | Process OB21 when the PC is switched on. |
| L | KF | 10 | Load ACCUM 1 with the factor 10. |
| T | RS | 96 | Transfer this value to system data word 96. |
| BE | | | |

**Caution**

The TIR, TDI, TBS and TNB operations are memory changing operations with which you can accesss the user memory and the system data area. These accesses are not monitored by the operating system. Improper use of the operations can lead to changes in the program and to a CPU crash.

## 8.3.3 Jump Operation

You can use a label to specify a jump destination within function blocks. For this jump operation, use a fixed-point number to indicate the jump displacement. Table 8-28 explains the "JUR" operation.

**Table 8-28. "JUR" Operation**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **JUR** | | | **Jump relatively**<br>Interrupt linear program scanning and continue at the point specified by the jump displacement. |
| | | **Parameter**<br>- 32768 to +32767 | |

* Not in the case of the CPU 941

**Processing the "JUR" Operation**

Execution of the "JUR" operation is independent of the RLO.
The parameter specifies the jump displacement directly. For example, parameter "1" means that processing will continue with the next one-word statement. Parameter "2" means processing will continue with the one-word statement directly after the next one-word statement.

Such labeling includes the following special features:

- The jump displacement is not corrected automatically. If changes are made in the Section of the program that is jumped over, the jump destination can be displaced.
- The jump destination should be in the same segment or block as the jump statement.

**Caution**

Avoid jumps over block boundaries since you have no control over the absolute location of blocks in the internal user memory.

## 8.3.4 Arithmetic Operation

An arithmetic operation increases the contents of ACCUM 1 by a specified value. The parameter represents this value as a positive or negative number. Table 8-29 shows the essential features of the "ADD" operation. An example follows the table.

**Table 8-29. Arithmetic Operation**

| Operation | Operand | | Meaning |
|---|---|---|---|
| **ADD** | | | **Add a constant**<br>Add byte or word constants. |
| **ID** | | **Parameter** | |
| | BN | | - 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 | BN | -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. |

## 8.3.5 Other Operations

Tables 8-30 and 8-31 provide an overview of the remaining system operations. Each table is followed by processing information.

**Table 8-30. Processing Operation**

| Operation | Operand | Meaning |
|---|---|---|
| **DI** | | **Process indirectly**<br>A formal operand indexes an operation; when the operation is executed, the block parameter is processed. Its number is in ACCUM 1. |

**Processing**

The "DI" operation works like the other processing operations. In contrast to "DO DW" or "DO FW", a formal operand is indexed for this operation. The statement that is executed by "DI" refers to the formal operand specified. However, the formal operand is not specified by its designation. You must load the "location number" of the formal operand in the parameter list into ACCUM 1 before the "DI" statement.

| Calling Block | Programmed FB | Explanation |
|---|---|---|
| : JU FB 2<br>NAME: PROCES<br>IN 0 : IW 10<br>IN 1 : IW 20<br>OUT : QW 100<br>.<br>.<br>. | NAME: PROCES<br>DECL : IN 0 IW<br>DECL : IN 1 IW<br>DECL : OUT QW<br>.<br>.<br>.<br>: L KF+2<br>: DI<br>: T QW 80 | ACCUM 1 is loaded with the constant "2".<br>The next statement is to process the formal operand that is in the second position in the parameter list.<br>The contents of IW 20 are transferred to output word QW 80. |

**Table 8-31. "TAK" and "STS" Operations**

| Operation | Operand | Meaning |
|---|---|---|
| **TAK** | | **Swap accumulator contents**<br>Swap the contents of ACCUM 1 and ACCUM 2 regardless of the RLO. The RLO and the condition codes are not affected. |
| **STS** | | **Stop immediately**<br>The CPU goes into the "STOP" mode regardless of the RLO. |

**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 cancelled manually (with the mode selector) or with the programmer function "PLC START".

## 8.4 Condition Code Generation

The processor of the S5-115U 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 (see Table 8-32). 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 8-32. 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 (see Table 8-33).

**Table 8-33. 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

### Condition Code Generation for Digital Logic Operations

Digital logic operations set CC 0 and CC 1. They do not affect the overflow condition code (see Table 8-34). The setting depends on the contents of the ACCUM after the operation has been processed.

**Table 8-34. Condition Code Settings for Digital Logic Operations**

| Contents of the ACCUM | 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 (see Table 8-35).
Code setting depends on the state of the last bit shifted out.

**Table 8-35. 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 (see Table 8-36). The state of the condition codes is based on the result of the conversion function.

**Table 8-36. Condition Code Settings for Conversion Operations**

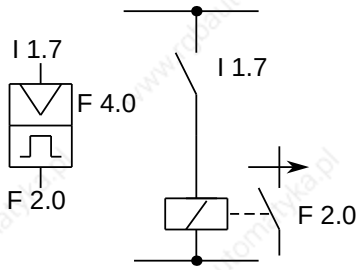
| Result after Arithmetic Operation is Executed | Condition Codes | | | Possible Jump Operations |
|---|---|---|---|---|
| | CC 1 | CC 0 | OV | |
| - 32768 * | 0 | 1 | 1 | JN, JM, JO |
| - 32767 to - 1 | 0 | 1 | 0 | JN, JM |
| 0 | 0 | 0 | 0 | JZ |
| +1 to +32767 | 1 | 0 | 0 | JN, JP |

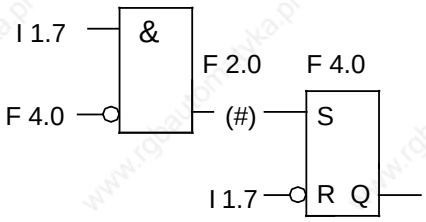
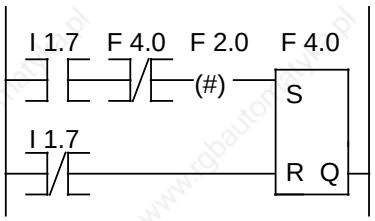
***** This number is the result of the conversion of KH=8000.

## 8.5 Sample Programs

Sections 8.5.1 through 8.5.3 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).

### 8.5.1 Momentary-Contact 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 2.0 ("edge flags").<br>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 2.0 is reset.<br>Therefore, flag F 2.0 is "1" for only one program run.<br>When input I 1.7 is switched off, flag F 4.0 is reset.<br>This resetting prepares the way for evaluation of the next leading edge of the signal at input I 1.7. |  |

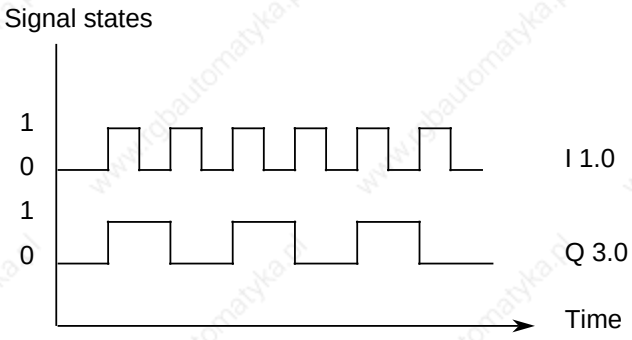
| STL | CSF | LAD |
|---|---|---|
| A I 1.7<br>AN F 4.0<br>= F 2.0<br>A F 2.0<br>S F 4.0<br>AN I 1.7<br>R F 4.0<br>**NOP 0** |  |  |

## 8.5.2 Binary Scaler

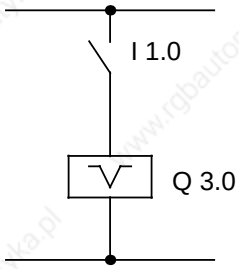
This section describes how to program a binary scaler.

**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**



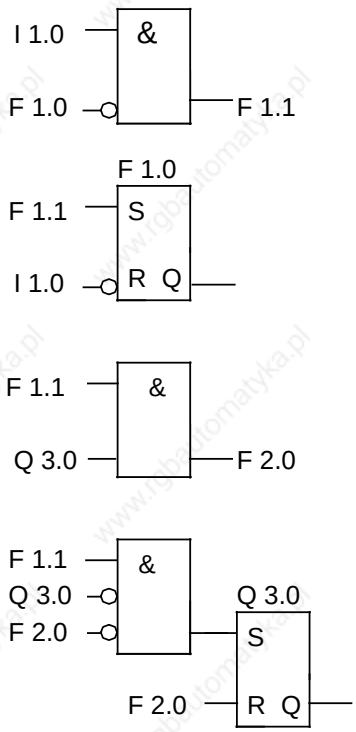

**Circuit Diagram**



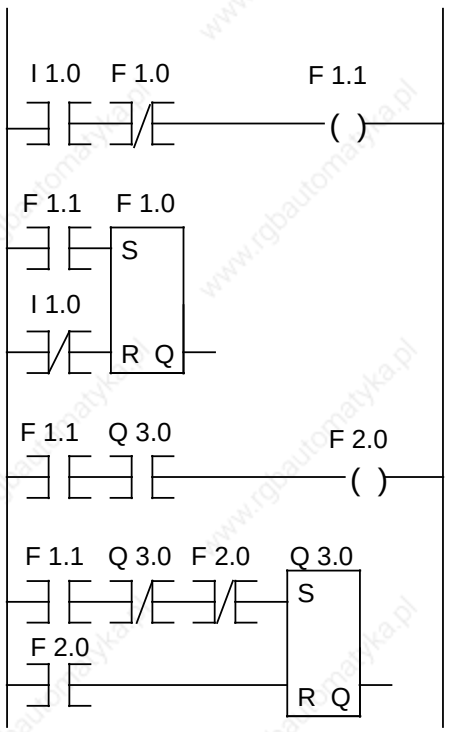

**STL**

```
A     I     1.0
AN    F     1.0
=     F     1.1
***
A     F     1.1
S     F     1.0
AN    I     1.0
R     F     1.0
NOP 0
***
A     F     1.1
A     Q     3.0
=     F     2.0
***
A     F     1.1
AN    Q     3.0
AN    F     2.0
S     Q     3.0
A     F     2.0
R     Q     3.0
NOP 0
```

**CSF**




**LAD**




**Note**

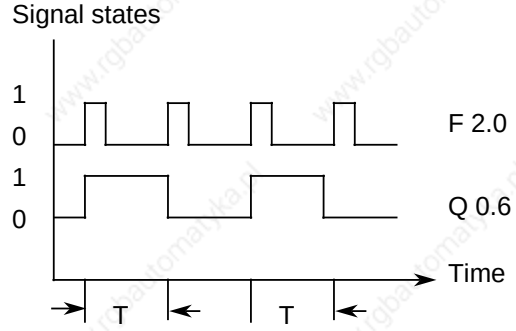
Output in CSF or LAD is possible only if you enter the segment boundaries "***" when programming in STL.

### 8.5.3 Clock (Clock-Pulse Generator)

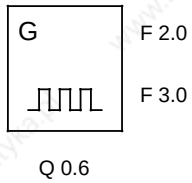
This subsection describes how to program a clock-pulse generator.

**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**



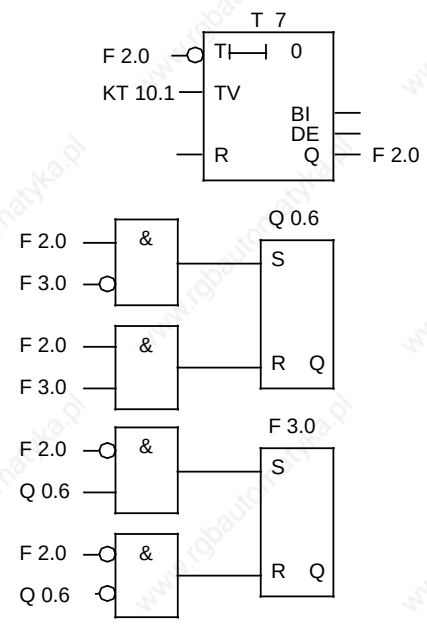

**Circuit Diagram**



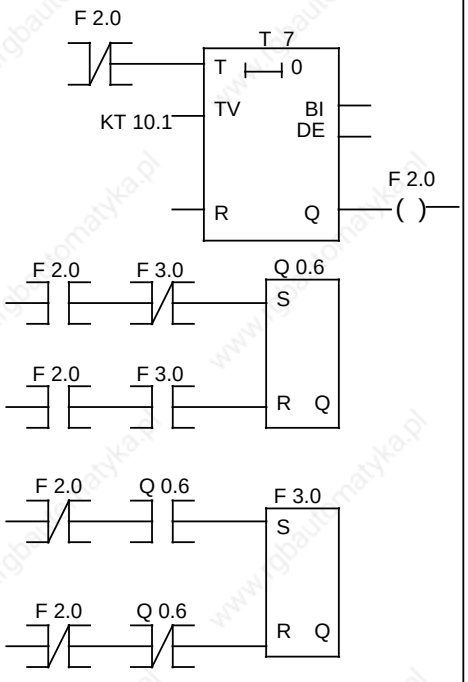

**STL**

```
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**



### 8.5.4 Delay Times

The following shows you how to program delay times with a timer in order to implement longer wait times.

| FB23 STL | Explanation |
|---|---|
| | LEN=23 |
| SHEET 1 | |
| SEGMENT 1 0000 | |
| NAME :WAIT | PROGRAMMED DELAY |
| | |
| 0005 :O F 0.0 | FORCE RLO "1" |
| 0006 :ON F 0.0 | |
| 0007 : | |
| 0008 :L KT 100.0 | ONE SECOND |
| 000A :SD T 0 | START TIMER |
| 000B SCHL :AN T 0 | LOOP |
| 000C :JU OB 31 | RESTART CYCLE TIME |
| 000D :JC =SCHL | |
| 000E :A T 0 | |
| 000F :R T 0 | RESET TIMER |
| 0010 :A T 0 | |
| 0011 :L KT 001.0 | RUN TIMER WITH RLO "0", |
| 0013 :SD T 0 | SO THAT NEW INITIATION |
| 0014 :BE | IS POSSIBLE |

You can use OB160 in the case of shorter times (up to 60 ms).

**Example:**

Program delay time of 30 ms:

```
L    KF  +30000
JU  OB  160
```

(see Section 11.2.2).

## 9 Interrupt Processing


9.1 Programming Interrupt Blocks . . . . . 9 - 1

9.2 Calculating Interrupt Response Times . . . . . 9 - 3

9.3 Process Interrupt Generation with the 434-7 Digital Input Module 9 - 5
9.3.1 Function Description . . . . . 9 - 5
9.3.2 Start-Up . . . . . 9 - 5
9.3.3 Initialization in Restart OBs . . . . . 9 - 5
9.3.4 Reading in the Process Signals . . . . . 9 - 7
9.3.5 Programming Example for Interrupt Processing . . . . . 9 - 8

9.4 Interrupt Processing with the Digital Input/Output Module 6ES5 485-7LA11 . . . . . 9 - 10
9.4.1 Function Description . . . . . 9 - 10
9.4.2 Operating the Module with Alarm Processing . . . . . 9 - 11
9.4.3 Operating the Module without Alarm Processing . . . . . 9 - 17
9.4.4 Notes on Characteristics of Inputs and Outputs . . . . . 9 - 18

**Figures**


9-1. Program for Interrupt OB (Principle) . . . . . 9 - 3
9-2. Position of the Mode Selector on the Module . . . . . 9 - 10
9-3. Address Assignments when Operating with "Alarm Processing" . . . . . 9 - 13
9-4. Example of Alarm Enable . . . . . 9 - 14
9-5. Address Assignments when Operating without "Alarm Processing" . . . . . 9 - 17


**Tables**


9-1. Additional Response Times . . . . . 9 - 4

# 9 Interrupt Processing

In this chapter you will learn the following:
- Which blocks are designed for handling process interrupts in the S5-115U
- How a process interrupt is intitiated
- What happens "internally" during interrupt processing
- The important points about simultaneous use of timed interrupts (timed-interrupt OBs)
- How to calculate the response times to a process interrupt.

In addition, Section 9.3 shows how to start up the 434-7 digital input module (with process interrupt).

## 9.1 Programming Interrupt Blocks

You can use interrupt-initiating modules in the S5-115U (e.g. signal preprocessing modules or the 434-7 digitial input module). These modules activate the CPU over an interrupt line in the I/O bus (S5 backplane bus). The CPU distinguishes between A, B, C or D interrupts depending on which interrupt line has been activated.
Each of these interrupts causes the operating system of the CPU to interrupt the cyclic or time-controlled program and to call an interrupt OB:
OB2 in the case of interrupt A (interrupt A is triggered by the 434-7 DI module, by some CPs or by IPs)
OB3 in the case of interrupt B (interrupt B is triggered by some CPs or by IPs)
OB4 in the case of interrupt C (interrupt C is triggered by some CPs or by IPs)
OB5 in the case of interrupt D (interrupt D is triggered by some CPs or by IPs)

**What Interrupts What and Where?**

As soon as the current cycle has been interrupted, i.e. while the CPU is processing an interrupt OB, further interrupts are automatically disabled. A running interrupt program can therefore not be interrupted.
If several interrupts are pending simultaneously, the priority is determined as follows:

| | |
|---|---|
| Highest priority: | Interrupt A |
| | Interrupt B |
| | Interrupt C |
| Lowest priority: | Interrupt D |

Interrupts A to D have higher priority than timed interrupts (OB10 to 13). The priority of OB6 in relation to interrupts A to D can be programmed (see Section 7.4.4).
An interrupt A, B, C or D interrupts the cyclic or time-controlled program after each operation. Exception: the TNB operation can be interrupted after every word in the case of CPUs 941, 942 and 943. The execution time of the TNB operation in the case of CPU 944 is so short that interruptibility has been dispensed with.
Integral function blocks and operating system routines can only be interrupted at specified points by an interrupt A (B, C, D) (cannot be influenced!).

If you have not programmed an interrupt OB, the cyclic or time-controlled program will continue immediately after the interrupt. If further interrupts occur during interrupt processing (edge is enough!), the CPU stores one of these interrupts per interrupt line! Prerequisite for detecting another interrupt is that the interrupt-initiating edges are separated by an interval of at least 12 µs! The subsequent order of processing the interrupts follows the interrupt priority described above.

**Example:**
While the CPU is processing OB2, interrupt B occurs and shortly afterwards, interrupt A.
**Result**:
After processing OB2, the CPU calls OB2 again (via interrupt A) and only then does it call OB3.

If part of your cyclic or time-controlled program is not to be interrupted, you must protect this part of the program from interrupt using the "IA" (inhibit interrupt) operation. At the end of this "protected" part of the program, interrupts must be enabled again using the "RA" operation. While interrupts are disabled, one interrupt can be stored per interrupt line!
Disabling interrupts is necessary, e.g. if you use integral data handling blocks both in the cyclic/time-controlled program and in the interrupt program: you must disable interrupts before each integral data handling block call in the cyclic/time-controlled program!

**Warning**

Many standard function blocks for IPs revoke the interrupt disable because they work internally with the IA and RA operations!
If you use these standard function blocks in restart or in an "interrupt-protected" part of the program, the relevant interrupt OBs may be called inadvertently!

**Notes on Avoiding Programming Errors**

- Note that the block nesting depth of 32 levels must not be exceeded even when calling interrupt OBs!
- If you use the same flags in the interrupt-processing program as in the cyclic program, you must save the contents of these flags at the beginning of the interrupt-processing program (e.g. in a data block); at the end of an interrupt-processing program, transfer the saved contents of the flags back to the relevant flag bytes (words).

**Enabling Interrupts in the Restart Program (OB21, OB22)**

If you want interrupt responses already at restart, you must enable the interrupts at the beginning of the restart OB with the "RA" operation. Otherwise, the interrupts will only come into effect after the restart OB has been processed.

Example of interrupt OB (OB2, OB3, OB4, OB5)

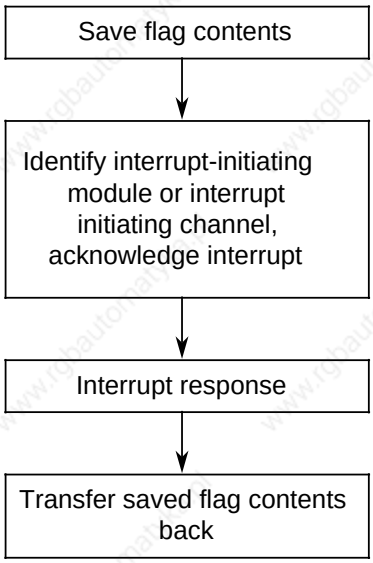



**Figure 9-1. Program for Interrupt OB (Principle)**

## 9.2 Calculating Interrupt Response Times

The total response time is the sum of the following:
- Signal delay of the interrupt-initiating module (=time from interrupt-initiating input signal change until activation of the interrupt line)
- Interrupt response time of the CPU
- Execution time of the interrupt program (=sum of all STEP 5 operations in the interrupt-evaluating program).

Calculate the interrupt response time of the CPU as follows:

Interrupt response time of the CPU=Basic response time+additional response times

The basic response time is 0.4 to 0.9 ms in the case of CPUs with one interface and 0.4 to 1.4 ms in the case of CPUs with two interfaces and applies when
- Integral FBs have not been used
- The integral clock has not been initialized
- PG/OP functions are not pending
- No computer link ((3964R) procedure) and no ASCII driver is active
- Time-controlled OBs are not programmed
  and
- SINEC L1 is not connected.

See Table 9-1 for the additional response times which can vary.

**Table 9-1. Additional Response Times**

| Additional CPU Functions Used | Delay of the Interrupt Response Time |
|---|---|
| Integral FBs*<br>Data handling blocks without data interchange | 0.5 ms |
| Data handling blocks with data interchange | 0.7 ms |
| Time-controlled OBs | 0.2 ms |
| Clock initialized | 0.5 ms |
| SINEC L1 LAN connected to SI 1 | 2.0 ms |
| SINEC L1 LAN connected to SI 2 | 0.5 ms |
| Computer link (3964(R)), ASCII driver | 0.5 ms |
| OP functions | 0.4 ms |
| Programmer functions<br>Force Var/Status Var<br>Status Block/Transfer Block<br>Display address**<br><br>Compress block with programmer or FB COMPR<br>- if no blocks are being shifted<br>- if blocks are being shifted | <br>2.4 ms screen loading (%)<br>0.5 ms<br>0.5 ms without rewrite<br>2.2 ms with rewrite<br><br>0.1 ms<br>19 ms per 1K statement of the block to be shifted |

* Cf. "Programmer functions, compress block" for FB238 (COMPR)

** If the programmer
- is connected to SI 1
  and
- accesses the I/O area,

the response time can increase to 240 ms.

## 9.3 Process Interrupt Generation with the 434-7 Digital Input Module

The 434-7 digital input module is an interrupt module with programmable interrupt generation.

### 9.3.1 Function Description

The process interrupts are processed in two different ways:
- Interrupt-initiating inputs can be identified by the control program.
- A yellow LED lights up on the module and a relay contact is closed (the relay contact can be accessed externally via the "MELD" outputs). This signal remains even in the event of power failure and can be reset by applying 24 V to the 24 V RESET input.

Although the 434-7 digital input module has only eight inputs, it occupies two bytes in the input I/O area and two bytes in the output I/O area, i.e. you can access two bytes of inputs and two bytes of outputs (input byte and output byte each have the same address). Because the 434-7 digital input module occupies two I/O bytes, the IM 306 has to be set to 16 channels for this module.
The addresses of the two consecutive I/O bytes occupied by the 434-7 are referred to in the following as "module address" and "module address+1".

- Use the two bytes of outputs in the restart OB for initializing the module (the "module address" byte indicates which input triggers the interrupt and the "module address+1" byte determines the type of the interrupt-initiating edge)
- You must use the two bytes of inputs when
  - you want to scan the status of the inputs (scan the "module address" byte)
  - you want to indentify inputs which have triggered the interrupt (scan the "module address+1" byte; only in interrupt program).

The status of the inputs must be scanned direct (L PY) since it is ***not*** transferred to the PII.

### 9.3.2 Start-Up

Assign a slot address to the module; the IM 306 interface module is to be set to 16 channels for the 434-7 digital input module!

### 9.3.3 Initialization in Restart OBs

The following must be programmed in the RESTART blocks OB21 and OB22:
- Which inputs are to trigger an interrupt
- Whether the interrupt is to be triggered by a rising or falling edge.

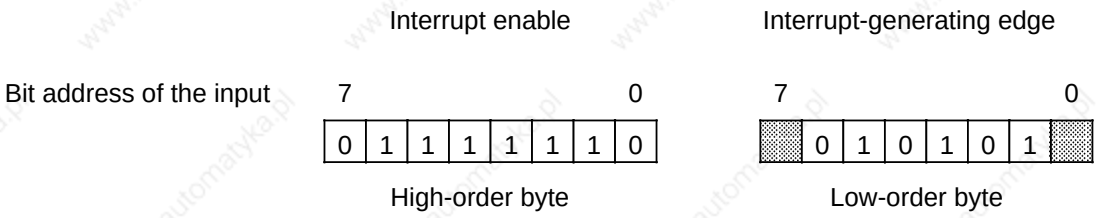
This information is stored in two bytes which the program in OB21 or OB22 transfers to the module.
In the "module address" byte, mark which inputs are to trigger an interrupt, and in the "module address+1" byte, mark which edge is to trigger the interrupt.

**Programming the RESTART Blocks**

| STL | | | Description |
|---|---|---|---|
| L | KM | a b | Load a two-byte bit pattern into ACCUM 1.<br>(a: Bit pattern of the interrupt enable; b: Bit pattern of the edge initiating the interrupt) |
| T | PW | x | Transfer the information from ACCUM 1 to the module<br>(x is the module start address). |

The bits in the high-order byte (byte a in this example) that was loaded into ACCUM 1 with the statement "L KM ab" correspond to the bit addresses of the eight input channels. If a bit is set to "1", the interrupt is enabled for this channel.
The bits in the low-order byte indicate whether the interrupt on this channel is triggered on a leading edge ("0") or on a trailing edge ("1").

**Example:** Triggering inputs 2, 4, and 6 on a leading edge. Triggering inputs 1, 3, and 5 on a trailing edge.



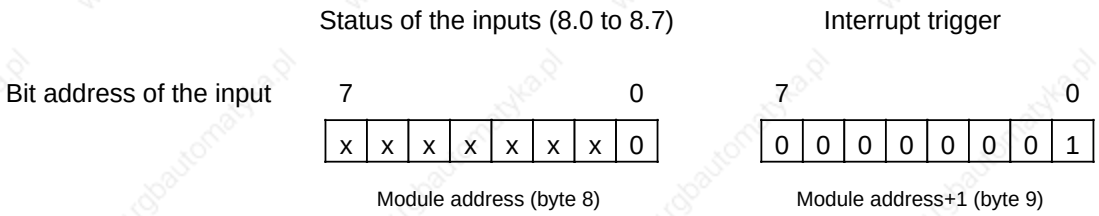

▨ = indicates irrelevant bits, since the corresponding bits in the high-order byte are set to "0" (no interrupt).

### 9.3.4 Reading in the Process Signals

The module offers a choice of two bytes for reading in the process signals:

- The "module address" byte reproduces the status of the inputs (regardless of whether the inputs have been initialized for interrupt processing).
- In the "module address+1" byte, the bits assigned to the interrupt-initiating input are set after an interrupt, regardless of the type of initiating edge! (The module has to be initialized at restart).

**Example:**
The 434-7 digital input module has starting address 8; it occupies I/O bytes 8 and 9. At startup, only bit 0 has been enabled for interrupt initiation. The interrupt is to be triggered by a falling edge. In the event of an interrupt, bytes 8 and 9 have the following values (provided the status of input 8.0 has not changed after edge change):




x=Status of the inputs (0 or 1)

There are two ways of evaluating the input signals with bytes 8 and 9:

- You can read the status of the inputs with direct I/O access (L PY 8) at any point in your control program. It is irrelevant whether the status of the inputs is read in the cyclic, time-controlled or interrupt-processing program.

- If you have initialized inputs at restart as interrupt-triggering inputs, you must program a specific interrupt response in OB2:
  - Acknowledge interrupt by reading the "module address+1" byte (in the example: byte 9; L PY 9)
  - Transfer the byte read to the PII (in the example: T IB 9)
  - Evaluate all inputs enabled for interrupt
  - Trigger interrupt response.

  After the byte "module address+1" (byte 9 in the example) has been loaded into the ACCUM, is automatically reset on the module! The module is therefore in a position to trigger another interrupt and so set another bit in this byte! This means that the "module address+1" byte can be read out only once after an interrupt in order to identify the "interrupt trigger".

### 9.3.5 Programming Example for Interrupt Processing

**Task**

A tray is to be accurately positioned at two points:
Position 1 is determined by terminating switch 12.
When the signal status of limit switch 1 changes from 0 to 1 (positive edge), drive 1 is to be switched off.
Position 2 is determined by limit switch 2.
When the signal status of limit switch 2 changes from 0 to 1 (negative edge), drive 2 is to be switched off.

The status of the limit switches is to be indicated by two LEDs:
LED 1 for "Signal status of limit switch 1"
LED 2 for "Signal status of limit switch 2"

**Implementation**

The 434-7 module has starting address 8. The IM 306 is set to 16 channels for the 434-7.
Limit switch 1 is assigned to channel 0 of the module, limit switch 2 is assigned to channel 1 of the module.

The OB21 and OB22 restart programs have the task of initializing the module:

| STL OB21/OB22 | Meaning |
|---|---|
| L KM 0000 0011 0000 0010<br><br>T PW 8<br><br>BE | Initialization of the interrupt inputs:<br>Enable channel 0: positive edge<br>Enable channel 1: negative edge |

The interrupts are evaluated in OB2:
Drive 1 is switched off by resetting output Q 0.0
Drive 2 is switched off by resetting output Q 0.1.

The status of the LEDs is updated in the cyclic program section:
When output Q 1.0 is set, LED 1 lights up
When output Q 1.1 is set, LED 2 lights up.

Evaluating the interrupt request in OB2:

| STL OB2 | Meaning |
|---|---|
| L PY 9<br>T IB 9 | Acknowledge interrupt by loading the "mod. addr. +1" byte; Transfer to PII |
| A I 9.0<br>R Q 0.0 | Scan: Did limit switch 1 trigger the interrupt?<br>If yes, reset output Q 0.0 (switch off drive 1) |
| A I 9.1<br>R Q 0.1 | Scan: Did limit switch 2 trigger the interrupt?<br>If yes, reset output Q 0.1 (switch off drive 2) |
| L QB 0<br>T PY 0<br>BE | Transfer updated output byte QB 0 direct to the output module (direct I/O access to minimize response time!) |

Updating the LED statuses in the cyclic program:

| STL OB1 | Meaning |
|---|---|
| :<br>L PY 8<br>T IB 8 | Load the status of the inputs (low byte)<br>Transfer low byte to PII |
| A I 8.0<br>= Q 1.0 | Transfer the status of limit switch 1 to the LED |
| A I 8.1<br>= Q 1.1<br>:<br>BE | Transfer the status of limit switch 2 to the LED |

Estimating the interrupt response time
(Prerequisite: no interrupts have been disabled with "IA")
The response time (i.e. the time between energizing the limit switch and switching off the drive) can be estimated as follows:

| | |
|---|---|
| | Signal delay of the 434-7 DI (approx. 1 ms) |
| + | Response time of the CPU (see Section 9.2) |
| + | Execution time of OB2 (=sum of all operation execution times) |
| = | Total response time |

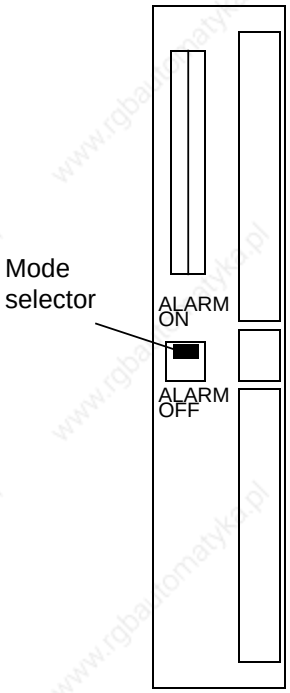
## 9.4 Interrupt Processing with the Digital Input/Output Module 6ES5 485-7LA11

The 485-7 digital input/output module is a 40-channel digital input/output module. The user can set and parameterize alarm generation.
Output current at "1" signal is 1.5 A per output.

### 9.4.1 Function Description

You can use the digital input/output module in two operating modes:
- As digital input/output module with alarm processing
- As digital input/output module without alarm processing

There is a mode selector on the back of the module for setting the modes. The "ALARM ON" position indicates "with alarm processing".




ALARM ON: Operation of the module with alarm processing
ALARM OFF: Operation of the module without alarm processing

**Figure 9-2. Position of the Mode Selector on the Module**

The digital input/output module occupies
- 4 input bytes
  and
- 4 output bytes

under a base address.
The input and output bytes are stored starting from the same base address.

You must set 32 channels for this module on the IM 306 interface module in the case of variable slot addressing.

The addresses of the contiguous I/O bytes occupied by the 485-7 module are indicated below with the addresses "x", "x + 1", "x + 2" and "x + 3"

### 9.4.2 Operating the Module with Alarm Processing

If you use the digital input/output module with alarm processing, you must set the mode selector on the back of the module to "ALARM ON" (see Figure 9.2).

The module can be used in the "with alarm processing" mode in all central controllers (CR 700-0/1/2/3) but not in the expansion units (ER 701-0/1/2/3).

When alarm processing is set, the digital input/output module gives you 4 input channels for alarm processing (see Figure 9.3).

Process alarms are always initiated by a "0" to "1" transition at the alarm inputs.

In addition, in the case of process alarms at the input (x + 2).4, the signal states at inputs x.0 to x.3 are stored in the alarm register (see Figure 9.3).

The time at which inputs x.0 to x.3 are stored corresponds to the time of storing the alarm input in the alarm register.
However, please note the different input delay times of the inputs:
- alarm inputs: typ. 1.5 ms
- "normal" inputs: typ. 3 ms

If a change at one of the inputs x.0 to x.3 is also to be stored, allowance must be made for the difference between the typical delay times of the inputs relative to the alarm input.

**Reading the inputs**

Input bytes x and x + 1 can always be read. These input bytes are accounted for in the process image of the inputs (access with A I, L IB, L IW, L PB, L PW, ...).

Input byte x + 2 can always be read except during restart. This byte is not accounted for in the process image of the inputs, i.e. input byte x + 2 must be accessed via direct I/O access (L PB).

**Response to an alarm request**

A reponse to an alarm request must take place in OB2:

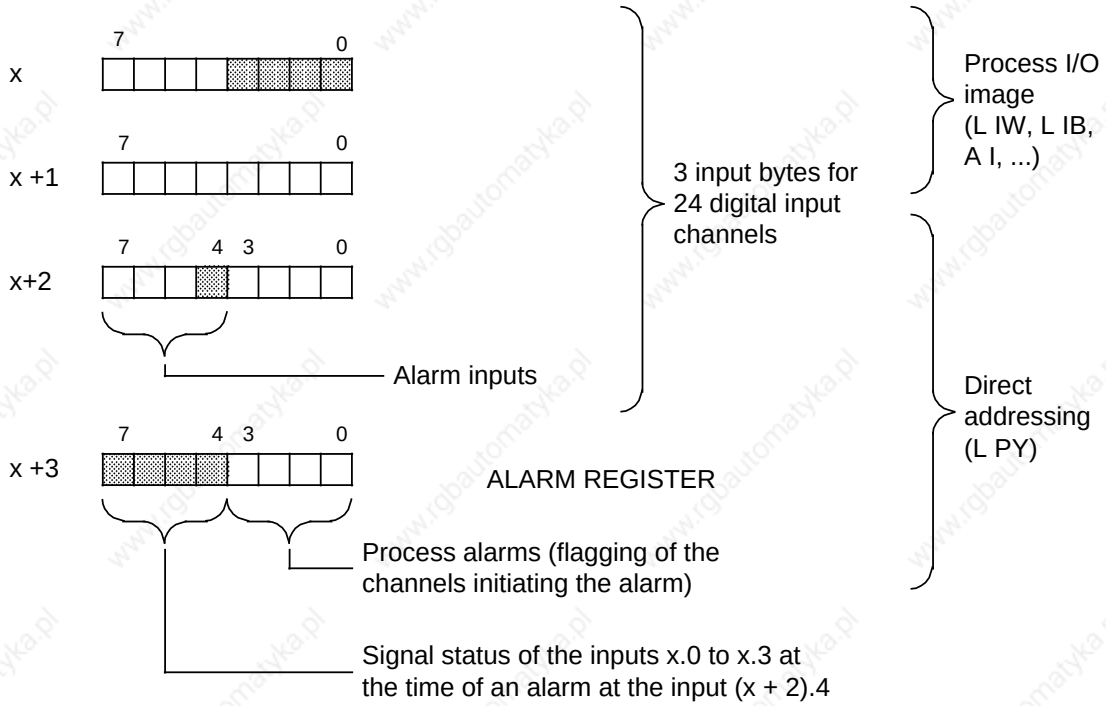
- Read the alarm register byte (x + 3) and transfer it to the process image of the inputs. Reading the alarm register causes bits 0 to 3 on the module to be deleted. This acknowledges the alarm and enables the module for further alarms.
- Bits 0 to 3 of the read alarm register show which of the 4 alarm inputs has initiated an alarm. All the bits for which the relevant alarm input has been enabled must be evaluated and the desired alarm response must be made.
- If bit 0 of the alarm register is set (alarm at input (x + 2).4), bits 4 to 7 of the alarm register show the status of inputs x.0 to x.3 at the time of initiation of the alarm.

**Alarm enable**

You must program the alarm enable in restart OBs OB21 and OB22. This is done by setting each bit under the output addresses (x + 2).0 to (x + 2).3 (see Figure 9.3).

**Address assignment when operating the module with alarm processing**

- 4 input bytes are assigned from the base address x.
  2 input bytes are stored in the process image; input bytes x + 2 and x + 3 must be addressed direct.




- 4 output bytes are assigned in the process I/O image from the base address x.

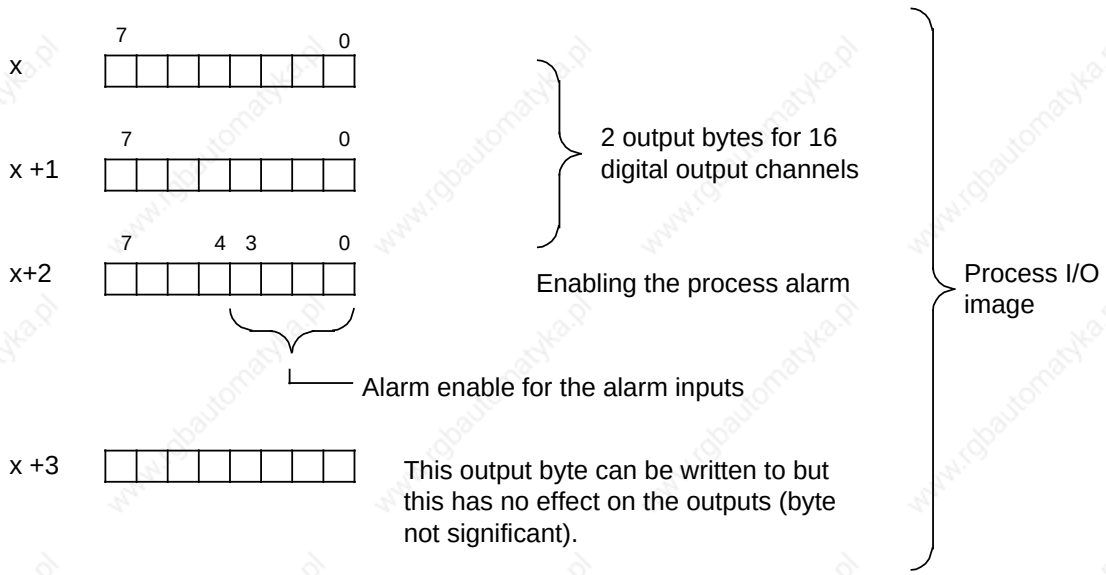



**Figure 9-3. Address Assignments when Operating with "Alarm Processing"**

**Example of alarm processing**

The module is operated with base address 0. You need 2 channels as alarm input. If you want to enable inputs 2.4 and 2.5 as alarm inputs, you must set outputs 2.0 and 2.1 in the restart OB (see Figure 9.4).

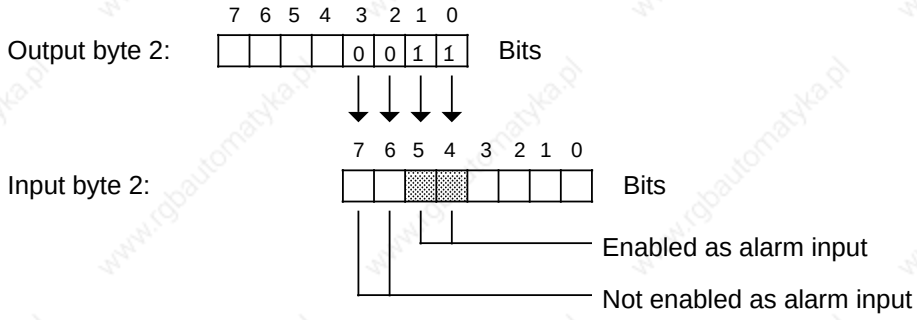



**Figure 9-4. Example of Alarm Enable**

The following programming example shows:
- How the module is parameterized at restart
- How alarm evaluation is handled in OB2 and
- How all inputs and outputs are accessed cyclically.

| OB 21 STL | Explanation |
|---|---|
| :<br>:<br>:<br>:<br>:L KM 00000000 00000011<br>:T PY 2<br>:BE | Alarm enable DIDQ485<br><br>Enabling of inputs 2.4 and 2.5<br>for alarm generation<br>(Set bit 0 and 1 in output byte 2) |

| OB 22 STL | Explanation |
|---|---|
| :<br>:<br>:<br>:<br>:L KM 00000000 00000011<br>:T PY 2<br>:BE | Alarm enable DIDQ485<br><br>Enabling of inputs 2.4 and 2.5<br>for alarm generation<br>(Set bit 0 and 1 in output byte 2) |

| OB 1 STL | Explanation |
|---|---|
| :<br>:<br>:L PY 2<br>:T EB 2<br>:<br>:<br>:<br>:<br>:<br>:<br>:L IW 0<br>:T QW 0<br>:L IB 2<br>:T QB 4<br>:BE | Cyclic program<br><br>Input byte 2 is not updated by the process<br>image transfer and must therefore be read<br>in with direct I/O access.<br><br>Program section with any evaluation of<br>inputs I 0.0 to I 2.7<br><br>Here:<br>Input 0 of the DIDQ 485 is output at<br>output word 0 of DIDQ 485 Input<br>byte 2 is flagged in output byte 4<br>(other module). |

| OB 2 STL | Explanation |
|---|---|
| :<br>:<br>:L PY 3<br>:T IB 3<br>:<br>:<br>:<br>:A I 3.0<br>:JC PB 1<br>:<br>:A I 3.1<br>:JC PB2<br>:<br>:BE | Alarm response and evaluation<br><br>Read alarm register and enter<br>in PII (acknowledge the alarm<br>with L PY3; this deletes bits<br>0 to 3 in the alarm register and<br>enables the module again for alarms)<br>I 3.1=0: alarm initiated by<br>input 2.4<br>(PB1: alarm response)<br>I 3.1=1: alarm initiated by<br>input 2.5<br>(PB2: alarm response) |

| PB 1 STL | Explanation |
|---|---|
| :<br>:<br>:<br>:<br>:L QB 6<br>:L KB 1<br>:+F<br>:T PY 6<br>:<br>:<br>:A I 3.4<br>:= Q 5.0<br>:A I 3.5<br>:= Q 5.1<br>:U E 3.6<br>:= Q 5.2<br>:A I 3.7<br>:= Q 5.3<br>:L QB 5<br>:T PY 5<br>:BE | Alarm response in the case of alarm<br>at input 2.4<br><br>Here:<br>Counting and flagging the<br>initiated alarms at input 2.4<br><br>Flagging the statuses of input I 0.0<br>to I 0.3 at the time of alarm at I 2.4<br>(statuses stored in bits 4 to 7 of the<br>alarm register)<br><br>Flagged in QB5<br><br><br><br><br>Direct access for faster<br>response time |

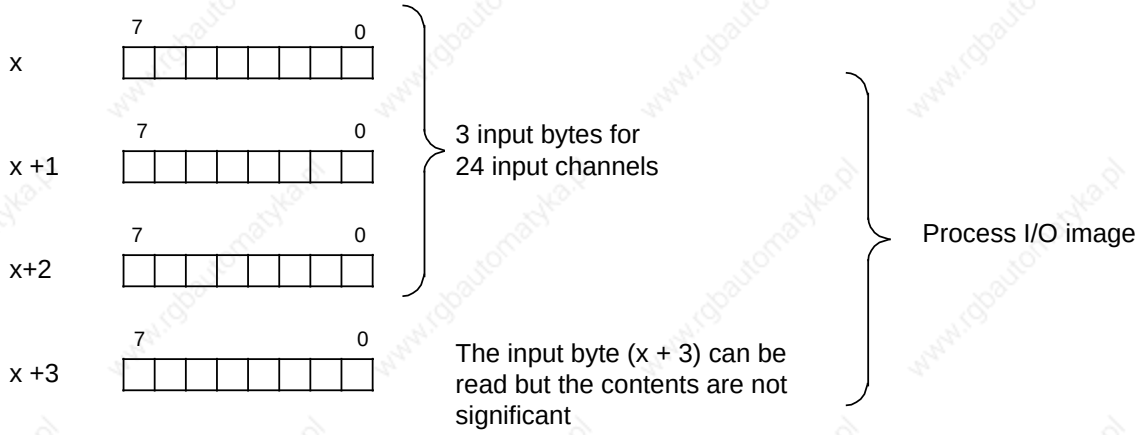
| PB 2 STL | Explanation |
|---|---|
| :<br>:<br>:<br>:<br>:L QB 7<br>:L KB 1<br>:+F<br>:T PY 7<br>:BE | Alarm response in the case of alarm<br>at input 2.5<br><br>Here:<br>Counting and flagging the<br>initiated alarms at input 2.5 |

### 9.4.3 Operating the Module without Alarm Processing

You can also use the module as a straightforward digital input/output module, in which case, the mode selector on the back of the module is set to the "ALARM OFF" position. When operating without alarm processing, you can use the module in all central controllers (CR 700-0/1/2/3) and in all expansion units (ER 701-0/1/2/3). Further parameterizing of the module is not required. You can address the module like any normal input/output module without restriction.

**Address assignments when operating the module without alarm processing**

- 4 input bytes are assigned in the process I/O image from the base address.




- 4 output bytes are assigned in the process I/O from the base address.

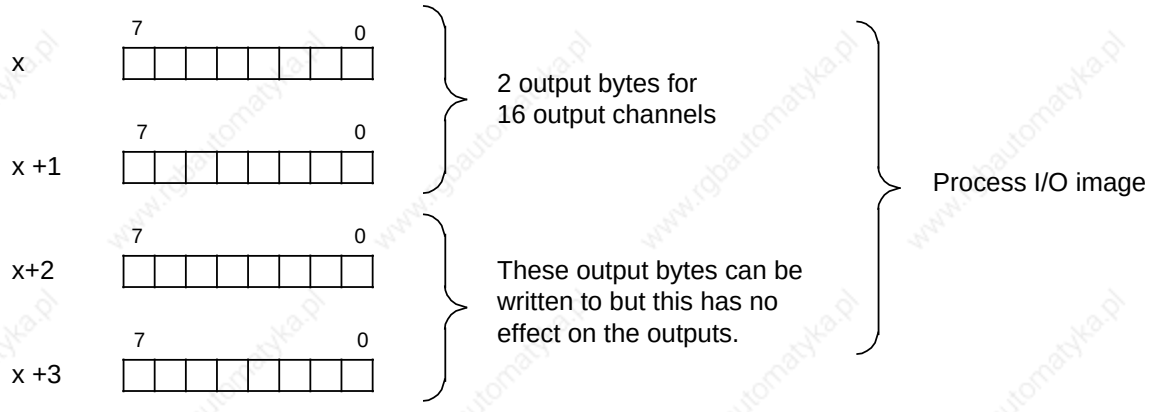



**Figure 9-5. Address Assignment when Operating "Without Alarm Processing"**

### 9.4.4 Notes on the Characteristics of Inputs and Outputs

The inputs and outputs of the module require an external load power supply.
The load current of all inputs and of output byte "x" is supplied through L1+.
The load current of output byte "x+1" is supplied through L2+.
The LEDs for the inputs and outputs are driven from the 5 V side of module.

This results in the following:

- An input signal is only detected and indicated via LED if the PLC power supply is switched on and the load voltage L1+ (at pin 36) is switched on.
- Output LEDs can indicate signal status 1 even if 24 V are not applied to the output. This is the case when there is no load voltage supply or if an output has short-circuited.

**Note**

If L1+ fails, the inputs are read in as logic "0" and the LEDs are switched off.

There is electronic short-circuit protection for the outputs. The maximum value of the tolerance-dependent switch-off current is 3.6 A. There is no overload protection if an output is operated in the range between 1.5 A (nominal value) and the maximum switch-off current.
You must ensure that in the event of cable short-circuit, the maximum switch-off current is exceeded. This can be done either by selecting suitable cable cross sections or by limiting the cable length.

## 10 Analog Value Processing


10.1 Principle of Operation of Analog Input Modules . . . . . 10- 1

10.2 Analog Input Module 460-7LA12 . . . . . 10- 3
10.2.1 Connecting Transducers to the 460-7LA12 Analog Input Module . 10- 4
10.2.2 Putting Analog Module 460-7LA12 into Operation . . . . . 10- 13

10.3 Analog Input Module 460-7LA13 . . . . . 10- 16

10.4 Analog Input Module 465-7LA13 . . . . . 10- 19
10.4.1 Connecting Transducers to the 465-7LA13 Analog Input Module . 10- 20
10.4.2 Starting Up the 465-7LA13 Analog Input Module . . . . . 10- 24

10.5 466-3LA11 Analog Input Module . . . . . 10- 28
10.5.1 Connecting Transducers to the 466-3LA11 Analog Input Module . 10- 29
10.5.2 Start-Up of the 466-3LA11 Analog Input Module . . . . . 10- 33

10.6 Representation of the Digital Input Value . . . . . 10- 39

10.7 Wirebreak Signal and Sampling for Analog Input Modules . . . . . 10- 51

10.8 Principle of Operation of Analog Output Modules . . . . . 10- 54
10.8.1 Connecting Loads to Analog Output Modules . . . . . 10- 56
10.8.2 Digital Representation of an Analog Value . . . . . 10- 58

10.9 Analog Value Matching Blocks FB250 and FB251 . . . . . 10- 60

10.10 Example of Analog Value Processing . . . . . 10- 64

## Figures


10-1. Block Diagram with Signal Interchange between the 460 Analog Input Module and the CPU . . . . . 10- 3
10-2. Pin Assignments for the 460 Analog Input Module . . . . . 10- 4
10-3. Connecting Transducers . . . . . 10- 5
10-4. Connecting Thermocouples . . . . . 10- 7
10-5. Connecting a Compensating Box to the Input of an Analog Input Module 10- 8
10-6. Connecting Resistance Thermometers (PT 100s) to a 460 Analog Input Module . . . . . 10- 9
10-7. Pin Assignments for Analog Input Modules . . . . . 10- 10
10-8. Connecting Transducers . . . . . 10- 11
10-9. Connecting Transducers (Four-Wire Transducer to a Two-Wire Range Card) 10- 12
10-10. Position of the Function Select Switches of the 460-7LA12 Analog Input Module (Rückseite der Baugruppe) . . . . . 10- 15
10-11. Verdrahtung der Geber bei AE 460-7LA13 (Klimabereich) . . . . . 10- 18
10-12. Block Diagram with Signal Interchange between a 465 Nonisolated Analog Input Module and the CPU . . . . . 10- 19
10-13. Pin Assignments for the 465 Analog Input Module . . . . . 10- 20
10-14. Connecting Resistance Thermometers (PT 100s) to a 465 Analog Module . 10- 22
10-15. Pin Assignments for Analog Input Module 465 . . . . . 10- 23
10-16. Position of the Function Select Switches of the 465-7LA12 Analog Input Module (Rückseite der Baugruppe) . . . . . 10- 26
10-17. Block Diagram of the 466-3LA11 Analog Input Module . . . . . 10- 28
10-18. Pin Assignments of the 466 Analog Input Module in the Case of Common-reference Measurement . . . . . 10- 29
10-19. Connecting Transducers to the 466 Analog Input Module (Common-reference Measurement) . . . . . 10- 30
10-20. Pin Assignments of the 466 Analog Input Module in the Case of Differential Measurement . . . . . 10- 31
10-21. Connecting Transducers to the 466 Analog Input Module (Differential Measurement) . . . . . 10- 32
10-22. Locations of the Mode Selectors on the 466-3LA11 Analog Input Module . 10- 33
10-23. Assignment of Switches S 1/S 2 to Channel Group . . . . . 10- 36
10-24. Representation of the Digitized Measured Value . . . . . 10- 39
10-25. PT 100 on SIMATIC Analog Input Modules . . . . . 10- 45
10-26. Block Diagram with Signal Interchange between CPU and a 470 Analog Output Module . . . . . 10- 55
10-27. Connecting Loads . . . . . 10- 56
10-28. Connecting Loads to Current and Voltage Outputs . . . . . 10- 57
10-29. Representation of an Analog Output Signal in Digital Form . . . . . 10- 58
10-30. Schematic Representation of Conversion . . . . . 10- 61
10-31. Example of Analog Value Processing . . . . . 10- 64
10-32. Function of the 460 Analog Input Module . . . . . 10- 65
10-33. Setting Mode Selectors I and II . . . . . 10- 65
10-34. Function of the 470 Analog Output Module . . . . . 10- 66

**Tables**


10-1. Range Cards . . . . . 10-.14
10-2. Setting Functions on the 6ES5 460-7LA12 Module . . . . . 10- 15
10-3. Setting Functions on the 6ES5 460-7LA13 Module . . . . . 10- 17
10-4. Range Cards . . . . . 10-.25
10-5. Setting Functions on the 6ES5 465-7LA12 Module . . . . . 10- 27
10-6. Setting the Type of Measurement (Common-reference or Differential) . . . 10- 33
10-7. Setting Current/Voltage Measurement for Channel Group I . . . . . 10- 34
10-8. Setting Current/Voltage Measurement for Channel Group II . . . . . 10- 34
10-9. Setting Current/Voltage Measurement for Channel Group I . . . . . 10- 35
10-10. Setting Current/Voltage Measurement for Channel Group II . . . . . 10- 35
10-11. Setting Current/Voltage Measurement for Channel Group III . . . . . 10- 35
10-12. Setting Current/Voltage Measurement for Channel Group IV . . . . . 10- 35
10-13. Setting the Measuring Range for One Channel Group (4 Channels per Group) . . . . . 10- 36
10-14. Setting the Data Format . . . . . 10- 37
10-15. Setting the Connection Type . . . . . 10- 38
10-16. Setting the Module Starting Addresses . . . . . 10- 38
10-17. Meaning of Bits 0 to 2 for Analog Input Modules . . . . . 10- 39
10-18. Representation of Digitized Measured Values of the 460 and 465 AI (Two's Complement; Measuring Range ±50 mV, ±500 mV, ±1000 mV) . 10- 40
10-19. Representation of Digitized Measured Values of the AI 460 and 465 (Two's Complement; Measuring Range ±5 V, ±10 V, ±20 mA) . . . . . 10- 41
10-20. Representation of Digitized Measured Values of the 460 and 465 AI (Number and Sign; Mesuring Range ±50 mV, ±500 mV, ±1000 mV) . . . . 10- 42
10-21. Representation of Digitized Measured Values of the 460 and 465 AI (Number and Sign; Measuring Range ±5 V, ±10 V, ±20 mA) . . . . . 10- 43
10-22. Representation of Digitized Measured Values of the 460 and 465 AI (Current Measuring Range 4 to 20 mA) . . . . . 10- 44
10-23. Representation of Digitized Measured Values of the 460 and 465 AI for Resistance-Type Sensors . . . . . 10- 45
10-24. Representation of Digitized Measured Values for PT 100 Climatic Measuring Range of the 460-7LA13 AI . . . . . 10- 46
10-25. Representation of Digitized Measured Values of the 466 AI (Measuring Range 0 to 20 mA; 0 to 5 V and 0 to 10 V; unipolar) . . . . . 10- 47
10-26. Representation of Digitized Measured Values (Two's Complement; Measuring Range ±5 V, ±20 mA and ±10 V; bipolar) . . . . . 10- 47
10-27. Representation of Digitized Measured Values (Number and Sign; Measuring Range ±5 V, ±20 mA and ±10 V; bipolar) 10- 47
10-28. Representation of Digitized Measured Values (Binary; Measuring Range ±5 V, ±20 mA and ±10 V; bipolar) . . . . . 10- 48
10-29. Representation of Digitized Measured Values (Measuring Range 0 to 1.25 V, and 0 to 2.5 V; unipolar) . . . . . 10- 48
10-30. Representation of Digitized Measured Values (Two's Complement; Measuring Range ±1.25 V, and ±2.5 V; bipolar) . . . 10- 48

**Tables**


10-31. Representation of Digitized Measured Values (Number and Sign; Measuring Range ±1.25 V, and ±2.5 V; bipolar) . . . . . 10- 49
10-32. Representation of Digitized Measured Values (Binary; Measuring Range ±1.25 V, and ±2.5 V; bipolar) . . . . . 10- 49
10-33. Representation of Digitized Measured Values (Measuring Range 4 to 20 mA and 1 to 5 V) . . . . . 10- 50
10-34. Wirebreak Signal in Conjunction with Resistance Thermometers . . . . . 10- 52
10-35. Scan Times . . . . . 10- 53
10-36. Analog Output Signals . . . . . 10- 59

# 10 Analog Value Processing

Analog input modules convert analog process signals to digital values that the CPU can process. Analog output modules perform the opposite function.

## 10.1 Principle of Operation of Analog Input Modules

The analog measured value is digitized and stored in a data register on the module. It can then be read and processed further by the CPU.

**Signal Interchange Between Module and CPU**

The CPU reads the digitized value from the module's memory via FB250 or a Load operation (L PW).
The complete measured value (2 bytes) is stored in CPU RAM.

**The 460, 465 and 466 Analog Input Modules**

Three different types of analog input modules are available:

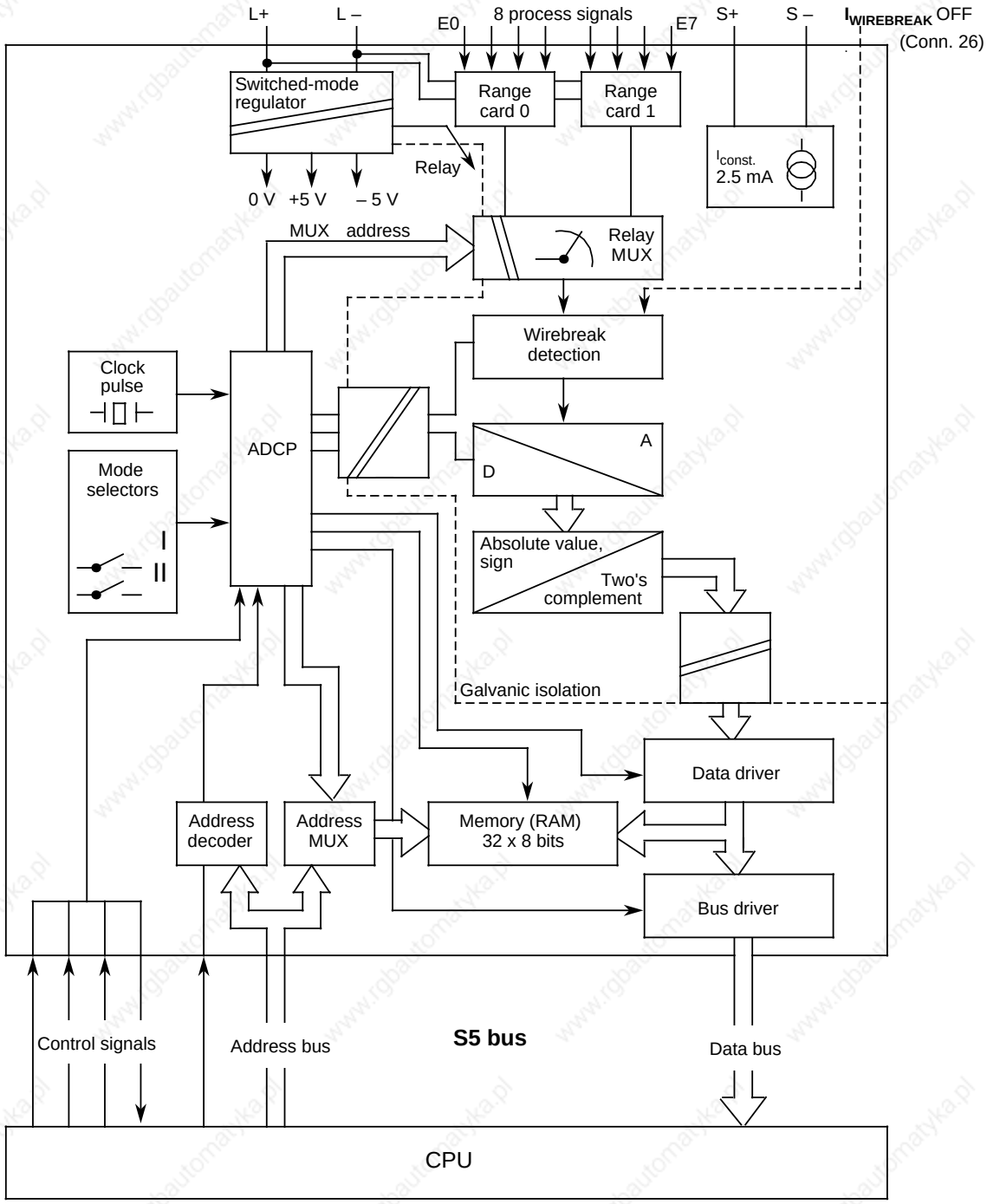
**6ES5 460-7LA12/7LA13**
- Galvanically isolated
- 8 channels
- 2 range cards
- Maximum permissible isolating voltage 60 V AC/75 V DC between a channel and ground as well as between channels.

**6ES5 465-7LA12**
- Non isolated
- 8/16 channels (selectable)
- 2/4 range cards
- 1 V max. permissible voltage between a channel and ground as well as between channels

**6ES5 466-3LA11**
- Floating
- 8/16 channels (switchable)
- Short coding times: 2 ms (8 channels) or 4 ms (16 channels)
- 12 different measuring ranges can be set using switches on the module
- Choice between common-reference measurement (16 channels) or differential measurement (8 channels)
- All operating modes can be set using switches on the module
- Maximum permissible isolation voltage $V_{ISO}$: 60 V AC/75 V DC; between the channels and ground (M) in each case; however, not between the channels themselves!

The block diagrams (Figures 10-1, 10-12 and 10-17) illustrate the method of operation as well as the signal interchange between the analog input modules and the CPU.
In the case of the **460** and **465** modules, a processor (ADCP) controls the multiplexer, analog-digital conversion and the forwarding of the digitized measured values to the memory or to the data bus of the programmable controller. The controller takes account of the module's operating mode, which is set at the relevant switch.
The process signals must be matched to the input level of the analog digital converter (ADC) to suit the application. You can match the signals with the 460 and 465 modules by plugging a suitable range card (voltage divider or shunt) into the receptacle on the frontplate of the analog input module.
In the case of the **466** module an internal controller handles all required functions.
You can adapt the process signals to the input level of the analog input modules in the case of the 466 module by specific settings of the measuring range switches.

## 10.2 Analog Input Module 460-7LA12

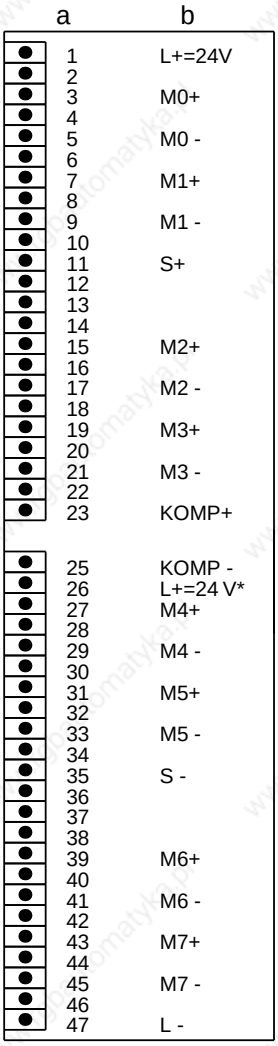



A/D Analog-digital converter (ADC)
ADUS ADU processor
MUX Multiplexer

**Figure 10-1. Block Diagram with Signal Interchange between the 460 Analog Input Module and the CPU**

## 10.2.1 Connecting Transducers to the 460-7LA12 Analog Input Module

**Pin assignments of the front connector**




460-7LA12

a=Pin No.
b=Assignment

*** Switching off the test current in the case of non-activated wirebreak signal**

**Figure 10-2. Pin Assignments for the 460 Analog Input Module**

Certain precautionary measures must be taken in order to make sure that potential difference $V_{CM}$ is not exceeded. Different measures are required for isolated and non-isolated transducers.

When isolated transducers are used, the measuring circuit can assume a potential to earth that exceeds the permissible potential difference $U_{CM}$ (refer to the maximum values for the various modules).
To prevent this, the transducer's negative potential must be connected to the module's reference potential (reference bus).

**Example:** Measuring temperature on a busbar with an isolated thermocouple.
In a worst-case situation, the measuring circuit can assume a potential that would destroy the module; this must be prevented through the use of an equipotential bonding conductor (see Figure 10-3).
Possible causes:
- Static charge
- Contact resistors through which the measuring circuit assumes the potential of the busbar (e.g. 220 V AC).

When using non-isolated transducers, the permissible potential difference $U_{CM}$ between the inputs and the reference bus must not be exceeded.

**Example:** Measuring the temperature of the busbar of an electroplating bath with a non-isolated thermocouple. The difference between the potential of the busbar and the reference potential of the module is max. 24 V DC. A 460 analog input module with floating input (permissible $U_{CM}$ is 60 V AC/75 V DC) is to be used.

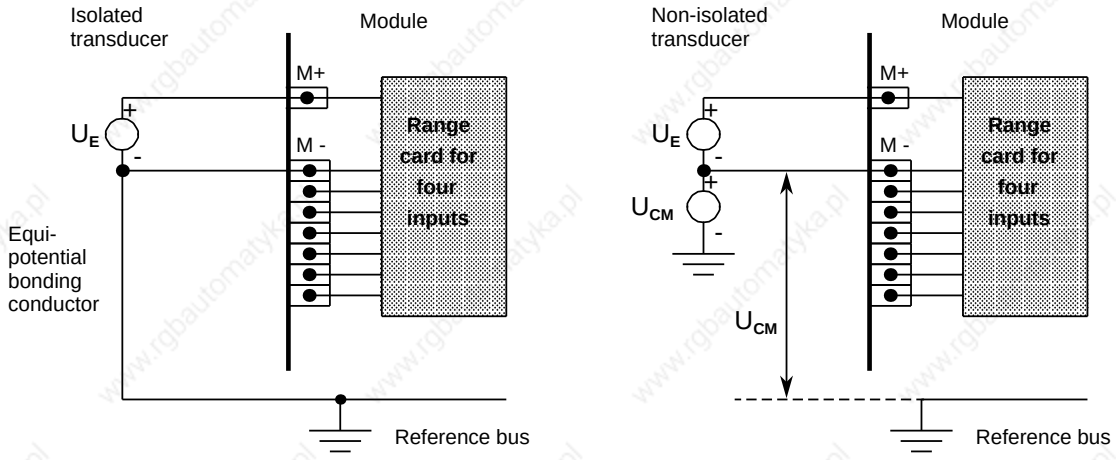



**Figure 10-3. Connecting Transducers**

You must observe various conditions when connecting current or voltage sensors to analog input modules, depending on what type of sensors are used.

**Note**

Detailed information on address assignment for analog modules is presented in Chapter 6 (Addressing/Address Assignments). Please observe the information regarding the overall structure (Section 3.4 in this manual).
Also observe the directions in Section 3.5 regarding shielding of the signal leads.

**Note**

Unused inputs must be terminated with a voltage divider or shunt (see Table 10-1).
In the case of the 498-1AA11 module, the unused inputs must be short-circuited (M+ with M - in each case).
Other modules require no additional wiring.
The galvanic isolation between the analog inputs and L+or L - is nullified when using the 498-1LAA51 module for a 2-wire transducer!

**Connecting Thermocouples with Compensating Box**

The influence of the temperature on the reference junction (in the terminal box, for instance) must be equalized using a compensating box. Please observe the following:

- The compensating box must have an isolated power supply.
- The power supply unit must have a grounded shielding winding.

Compensate as follows when all thermocouples connected to the module's inputs have the same reference junction:

- Provide a separate compensating box for each analog input module
- Bring the compensating box into thermal contact with the terminals
- Apply compensating voltage to pins 23 and 25 (KOMP+and KOMP -) on the analog input module (Figure 10-4)
- Set Function Select switch II on the module for operating a compensating box (see also Table 10-2)

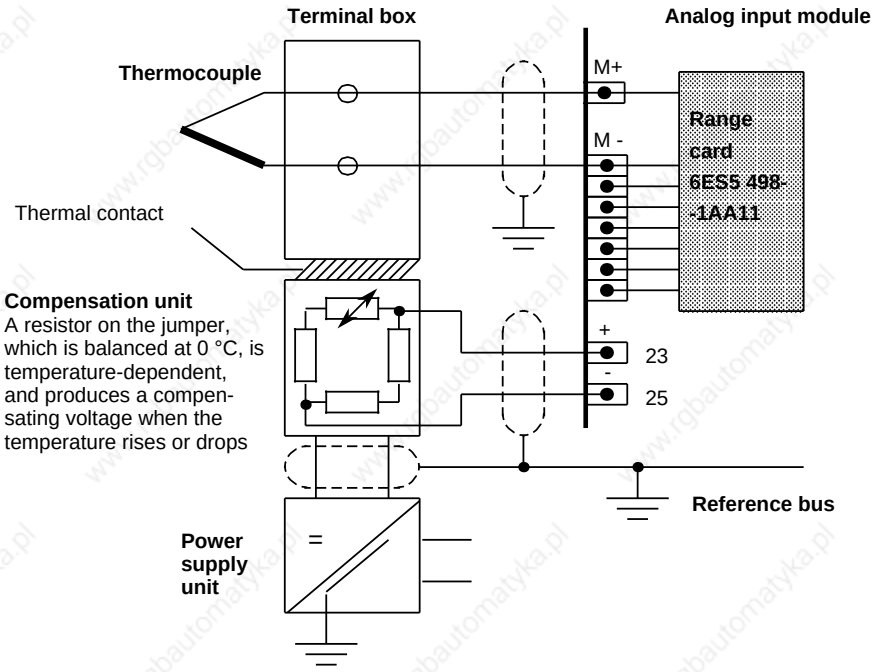



**Figure 10-4. Connecting Thermocouples**

Detailed information on thermocouples and compensating boxes can be found in Catalog MP 19.

When several thermocouples are distributed over areas with different temperature ranges, it is often advantageous to acquire different reference junction temperatures. In this case, the central compensating input is no longer used. A separate compensating box is used for each analog input channel to be compensated. KOMP+and KOMP - remain unconnected.

- Connect the relevant thermocouple in series with the compensating box.
- Run the remaining terminal leads from compensating box and thermocouple to the analog module (terminal M+and M - see Figure 10-5).
- Set Function Select switch II on the module to "Without reference junction compensation".

Compensation, i.e. correction of the temperature error, subsequently takes place in the compensating box rather than on the module.
The corrected value is thus available at terminals M+and M - of the relevant analog input channels, and is then converted into a digital value.

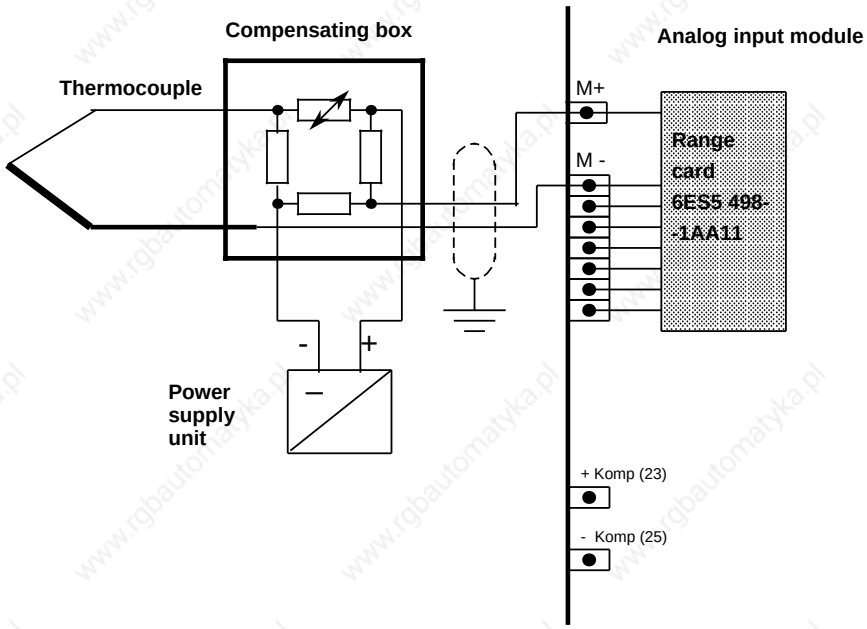



**Figure 10-5. Connecting a Compensating Box to the Input of an Analog Input Module**

**Connecting Resistance Thermometers (e.g. PT 100) with 6ES5 460-7LA12**

A constant-current generator supplies the series-connected resistance thermometers (max. 8 PT 100s) with a current of 2.5 mA over pins "S+" and "S -".

If you use the 498-1AA11 submodule, you must terminate the unused input channels with a short-circuiting jumper (see Figure 10-6, range card 2; channels 5 and 6).

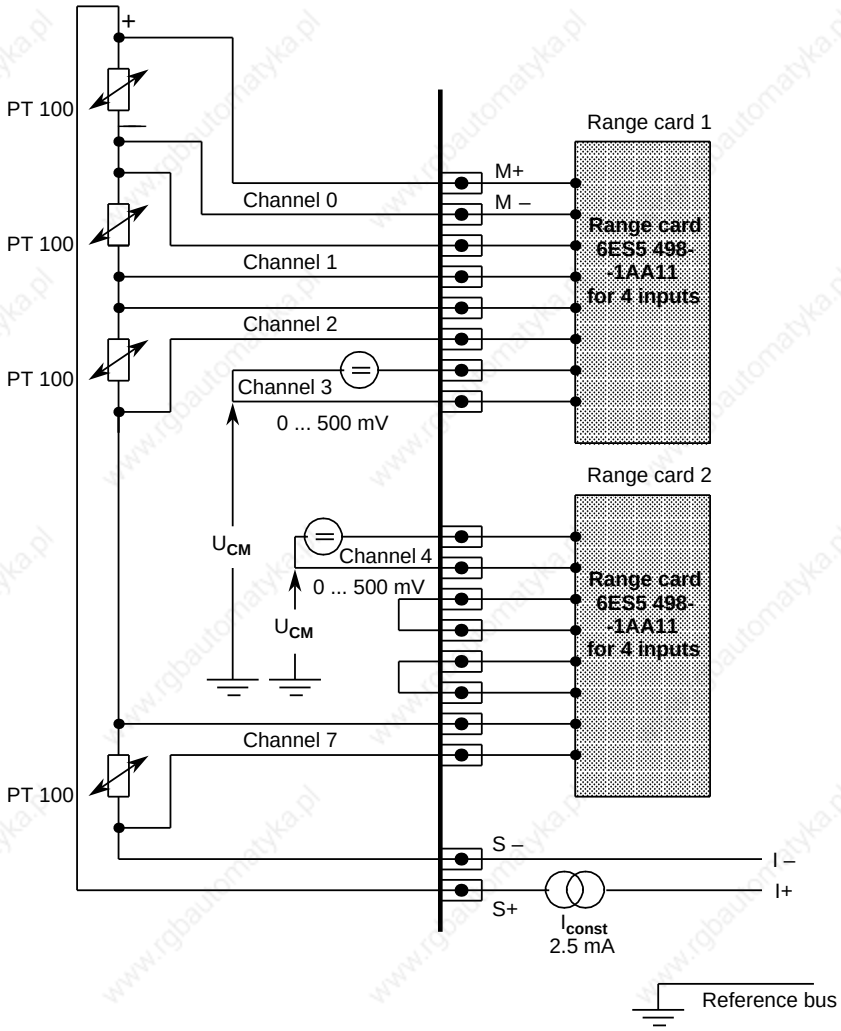
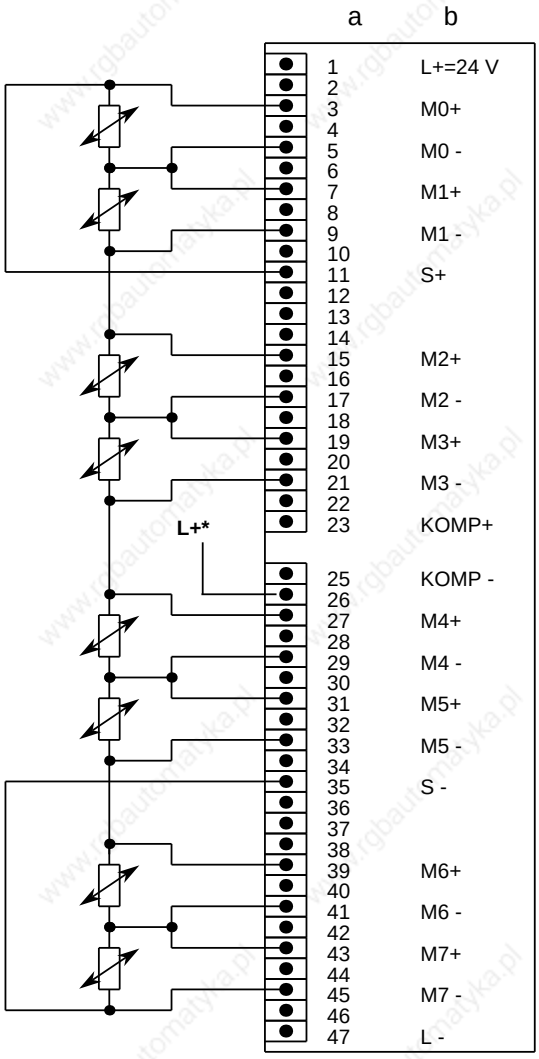



**Figure 10-6. Connecting Resistance Thermometers (PT 100s) to a 460 Analog Input Module**

If no PT 100 is connected to input channels 4 to 7, other voltages and currents can be measured on these channels using range cards 498-1AA21, -1AA31, -1AA41, -1AA51, -1AA61 or -1AA71.

The diagram below shows the pin assignments for resistance thermometers used on analog input module 460.




6ES5 460-7LA12

a=Pin no.
b=Assignment

* Required only for disconnecting the test current when the wirebreak signal is not activated

**Figure 10-7. Pin Assignments for Analog Input Modules**

**Connecting Transducers with Module 460-7LA12**

The inherently short-circuit-proof supply voltage is fed to the two-wire transducer over the range card.
Four-wire transducers have a separate power supply.

The diagram below shows how to connect two-wire and four-wire transducers.

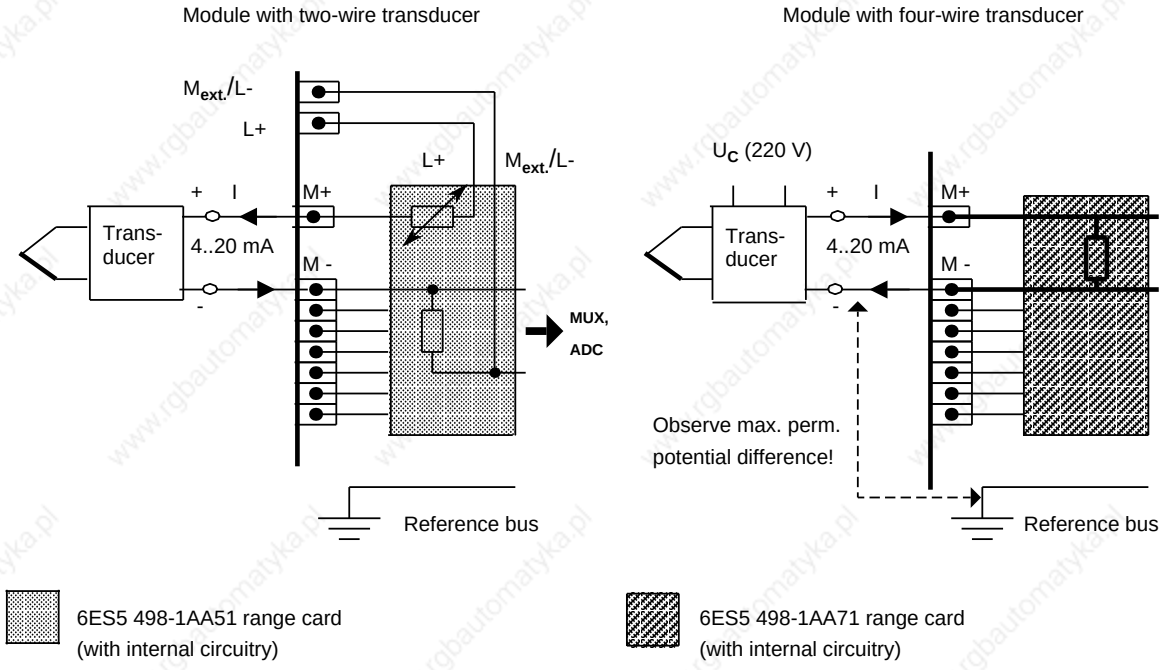
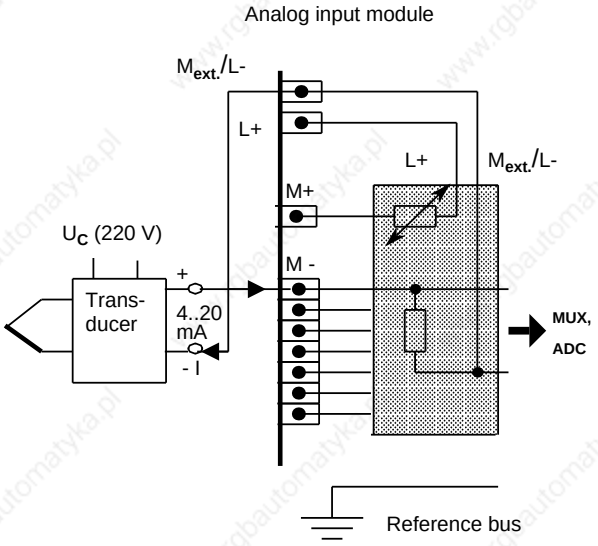



**Figure 10-8. Connecting Transducers**

The diagram below shows how to connect a four-wire transducer to a two-wire transducer range card (498-1AA51).




6ES5 498-1AA51 range card
(with internal circuitry)

**Figure 10-9. Connecting Transducers (Four-Wire Transducer to a Two-Wire Range Card)**

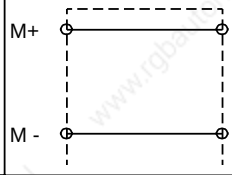
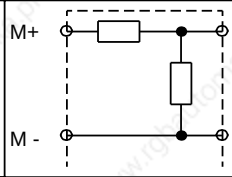
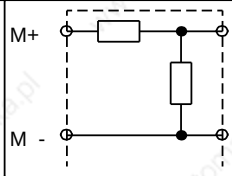
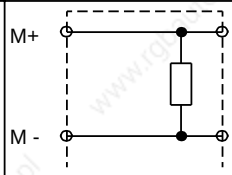
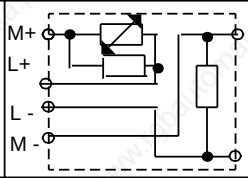
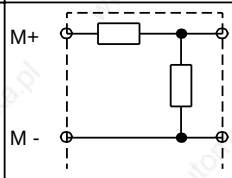
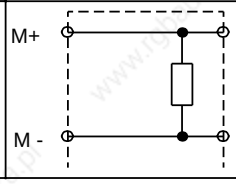
## 10.2.2 Putting Analog Module 460-7LA12 into Operation

Voltage dividers or shunt resistors can be plugged into the input modules as cards (see Table 10-1).
They match the process signals to the input level of the module.
These cards make it possible to set different measuring ranges.

**Connecting Range Cards**

Two range cards can be plugged into the 460 analog input module. One card specifies the measuring range of four inputs.
We offer voltage dividers, shunts and through-connection cards (see Table 10-1).

**Table 10-1. Range Cards**

| Range card 6ES5 498- | Circuitry (4 times each) | Function 500 mV/mA/PT100 | Function 50 mV |
|---|---|---|---|
| - 1AA11 |  | ±500 mV;<br>PT 100 | ±50 mV |
| - 1AA21 |  | ±1 V | ±100 mV * |
| - 1AA31 |  | ±10 V | ±1 V * |
| - 1AA41 |  | ±20 mA | ±2 mA * |
| - 1AA51** |  | +4 ...+20 mA<br>two-wire<br>transducer | |
| - 1AA61 |  | ±5 V | ±500 mV * |
| - 1AA71 |  | +4 ...+20 mA<br>four-wire<br>transducer | |

* Possible measuring range for "50 mV" setting, but with higher incidence of error

** When a -1AA51 range card is used, there is no longer any galvanic isolation between analog inputs and L+!

**Note**

Jumpers must be set in the front connector in the case of through-connection card 1AA11. Unused inputs need not be short-circuited in the case of voltage dividers or shunts.

You can set various functions on an input module by setting the Function Select switches on the rear of the module accordingly (see Table 10-2).

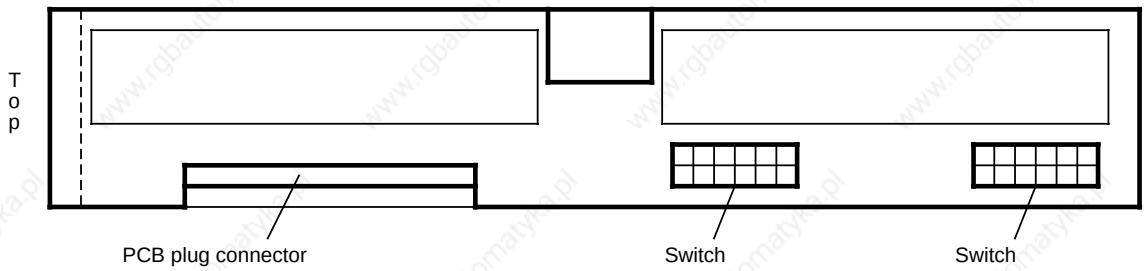



**Figure 10-10. Position of the Function Select Switches of the 460-7LA12 Analog Input Module**

> **Note**
>
> Selection of a function entails the setting of all switches.

**Table 10-2. Setting Functions on the 6ES5 460-7LA12 Module**

| Function | Setting on Switch | | Setting on Switch | |
|---|---|---|---|---|
| Reference junction compensation | | | Yes | No |
| Measuring range* (nominal value) | | | 50 mV | 500 mV |
| Analog value representation | | | Two's complement | Abs. value and sign |
| Sampling | Cyclic | Selective | | |
| System frequency | 50 Hz | 60 Hz | | |
| Wirebreak signal | Channel 0 to 3 | Channel 4 to 7 | | |
| No wirebreak signal | Channel 0 to 3 | Channel 4 to 7 | | |

* Setting for PT 100: Measuring range 500 mV

## 10.3 460-7LA13 Analog Input Module

The 460-7LA13 analog input module has been developed from the 460-7LA12 analog input module. It offers the following advantages:

- Lower power consumption and heating
- Lower weight
- New PT 100 climatic measurement range (-100 °C to +100 °C) with high resolution (1/40 °C)

All functions of the 460-7LA12 module are also available on the 460-7LA13 module.

The following features are identical with the 460-7LA12 module:

- Transducer cabling
- Use of the 6ES5 498 range cards
- Assignment of the front connector
- System behavior

Compared to the 460-7LA12 analog input module, the following features of the 460-7LA13 analog input module are new or different:

- New PT100 climatic measuring range (as alternative to the previous PT100 measuring range)
- Setting of the mode selector switches for the PT100 measuring ranges
- Setting of the mode selector switched for the measuring range 50 mV (e. g. for connection of thermocouples).

**New PT100 Climatic Measuring Range**

The same measuring range as with the 460-7LA12 analog input module exists also on the 460-7LA13 analog input module; i.e. the PT100 temperature range (-200 °C to +850 °C) is resolved in this measuring range to approximately 4000 units. This corresponds to a resolution of approx. 0.25 °C.

If the new PT100 climatic measuring range is selected via the mode selector switches, all eight analog inputs can be used in this measuring range only.

Do not use other than the 6ES5 498-1AA11 (50 mV/0.5 V) range card.

The following must be observed in conjunction with wire break monitoring in the PT 100 climatic measuring range:
If a line of the auxiliary circuit (IC+, IC-) is interrupted, the value "negative end value" is encoded for all inputs and the overflow bit is set to "1". In the case of transducer or measuring line break, the error bit for the corresponding channel is additionally set to "1".

For an exact representation of measured values in the PT100 climatic measuring range refer to Table 10.23.

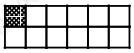
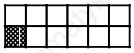
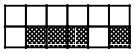
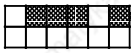
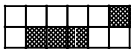
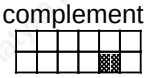
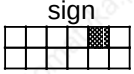
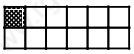
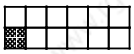
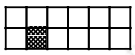
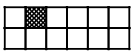
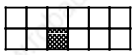
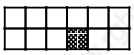
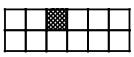
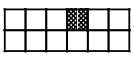
For setting of the "PT100" mode, the following markings are printed on the cover of the module:

Standard range: "resistance thermometer uncompensated full range"
Climatic measuring range: "resistance thermometer uncompensated low range"

**Setting of Mode Selector Switches I and II**

The mounting position and the setting of the mode selector switches corresponds to that of the 460-7LA12 module. The only difference is the setting of the PT100 measuring ranges (see Table 10.3).

**Table 10.3 Setting Functions on the 6ES5 460-7LA13 Module**

| Function | Setting on Switch | | Setting on Switch | |
|---|---|---|---|---|
| Reference junction compensation | | | Yes  | No  |
| Measuring range* (nominal value) | | | resistance thermometer compensated low range  <br>500 mV, V ... mA resistance thermometer uncompensated full range  <br>50 mV  | |
| Analog value representation | | | Two's complement  | Abs. value and sign  |
| Sampling | Cyclic  | Selective  | | |
| System frequency | 50 Hz  | 60 Hz  | | |
| Wirebreak signal | Channel 0 to 3  | Channel 4 to 7  | | |
| No wirebreak signal | Channel 0 to 3  | Channel 4 to 7  | | |

### Transducer Wiring

Transducers are wired in the same way as with the 460-7LA12 module. Unused inputs must be connected in parallel with switched inputs. An example is given in Fig. 10.11.

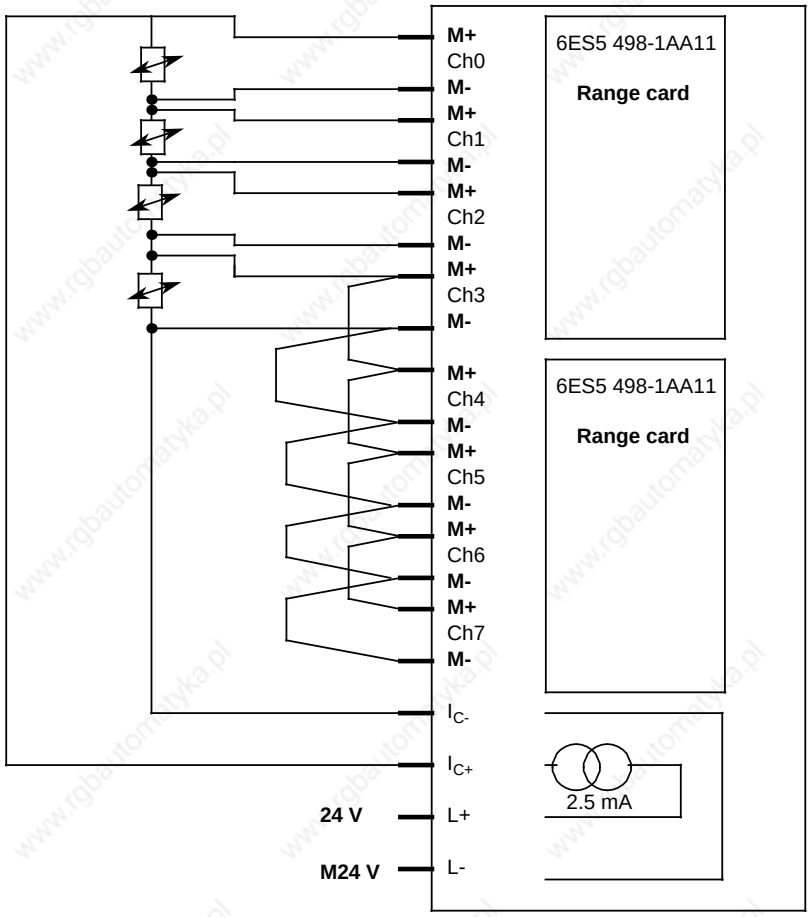
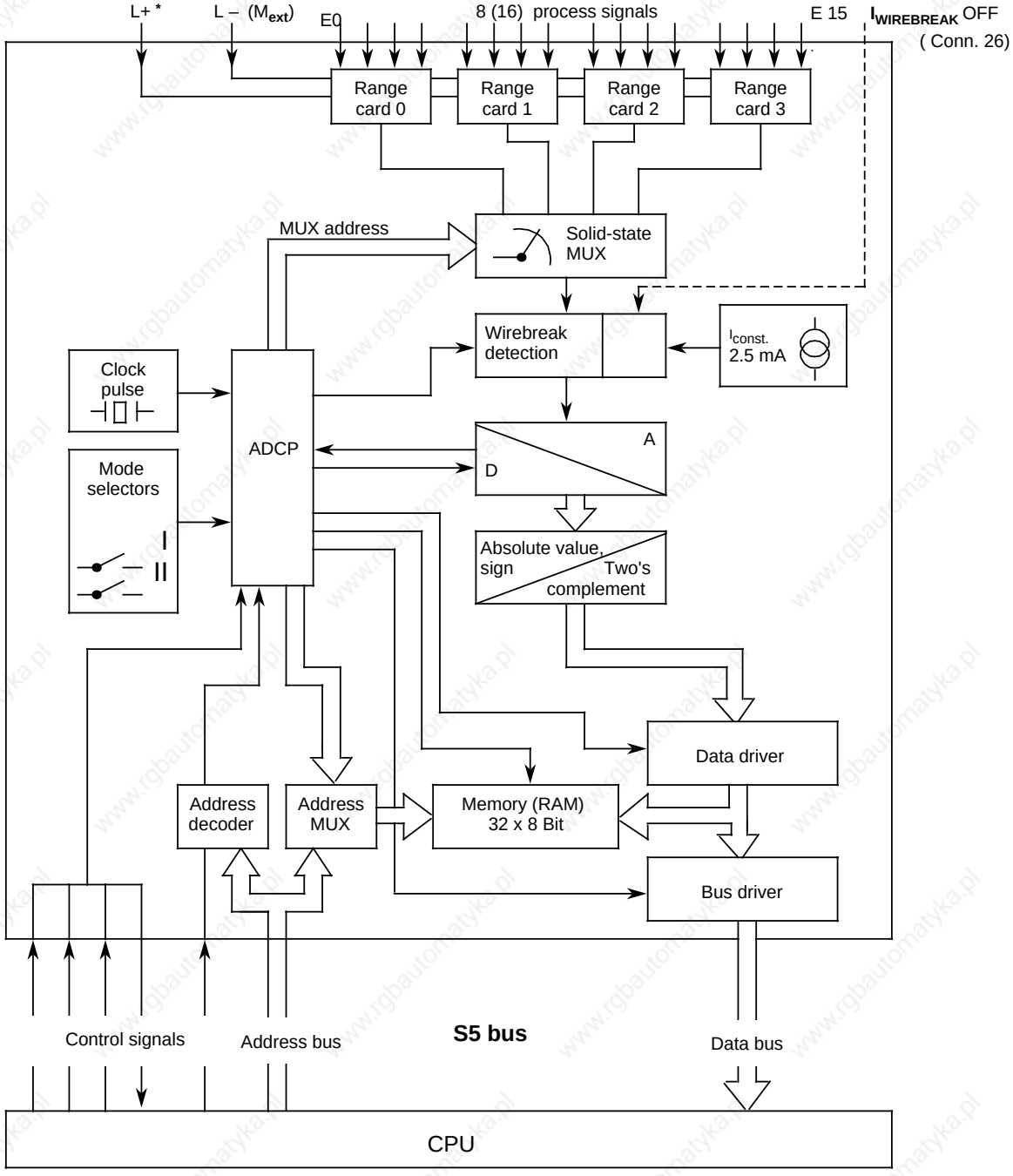



**Figure 10-11. Wiring of Transducers on the 460-7LA13 Analog Input Module (for Climatic Measuring Range)**

## 10.4 Analog Input Module 465-7LA13



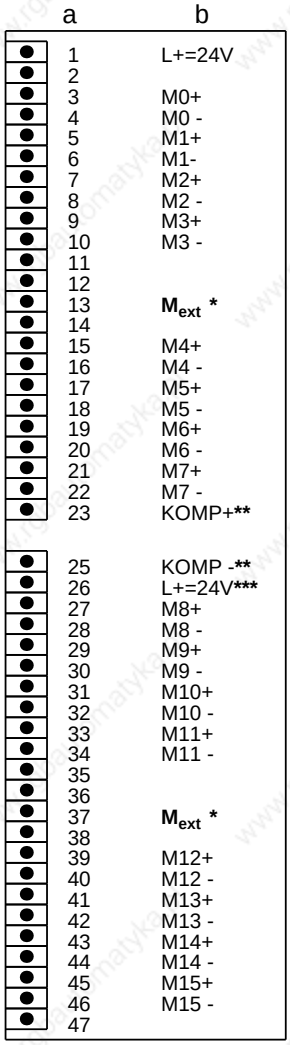

A/D Analog-digital converter (ADC)
ADCP ADC processor
MUX Multiplexer

* Required only when using a -1AA51 module

**Figure 10-12. Block Diagram with Signal Interchange between a 465 Non-Isolated Analog Input Module and the CPU**

## 10.4.1 Connecting Transducers to the 465-7LA13 Analog Input Module

**Pin assignments of the front connector**




465-7LA13

a=Pin No.
b=Assignment

* Connection to the central grounding point of the controller
** Connection of the compensating box
*** Switching off the test current in the case of non-activated wirebreak signal

**Figure 10-13. Pin Assignments for the 465 Analog Input Module**

**Note**

Connection of transducers is described in detail in Section 10.2.1.

**Note**

Unused inputs must be short-circuited when using the 6ES5 498-1AA11 through-connection card.

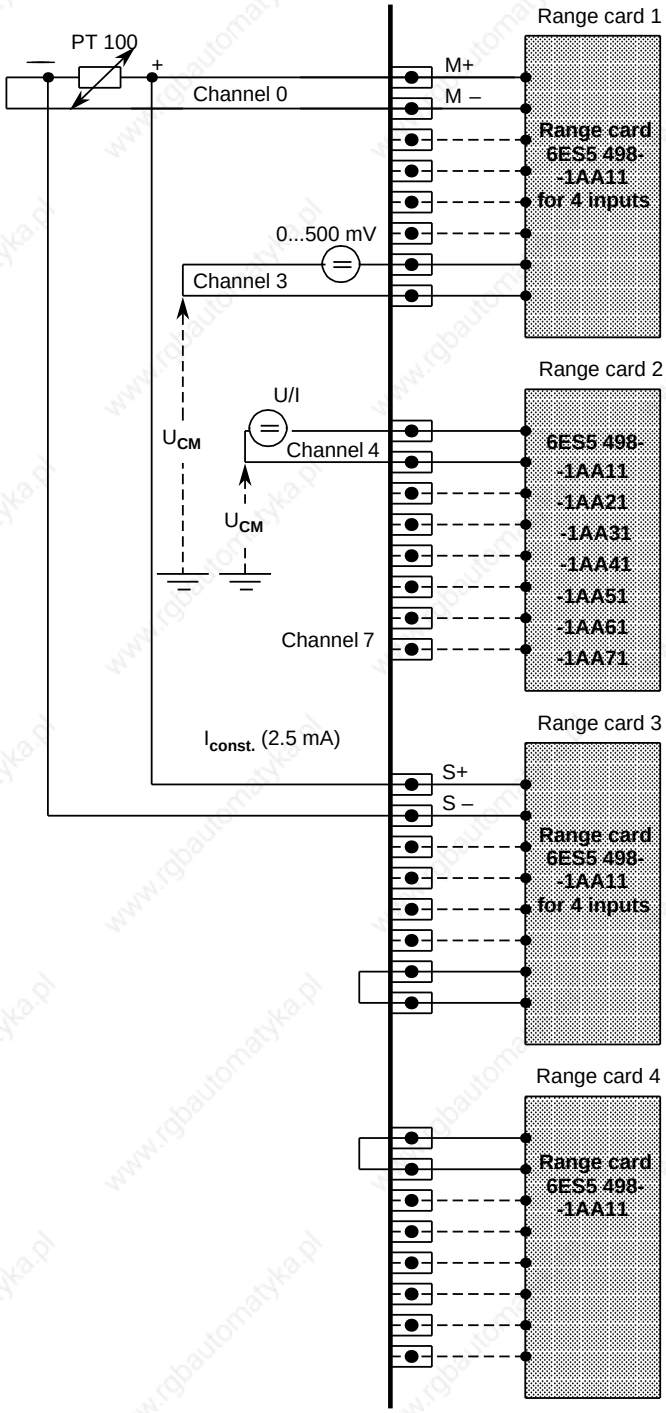
**Note**

Detailed information on address assignment for analog modules is presented in Chapter 6 (Addressing/Address Assignments). Please observe the information regarding the overall structure (Section 3.4 in this manual).
Also observe the directions in Section 3.5 regarding shielding of the signal leads.

**Connecting Thermocouples with Compensation Boxes**

Connection of thermocouples is the same as for the 460 module (see Section 10.2.1)

**Connecting Resistance Thermometers (PT 100) to a 465-7LA13 Analog Module**




A constant-current generator supplies the relevant resistance thermometer with a current of 2.5 mA over pins "S+" and "S -" via a range card (6ES5 498-1AA11) (see Figure 10-14).

The voltage on the PT 100 is picked off over inputs "M+" and "M -".

Other potential-free voltage sensors (500 mV voltage range) can be connected to those inputs (M+/M -) not used for resistance thermometers.

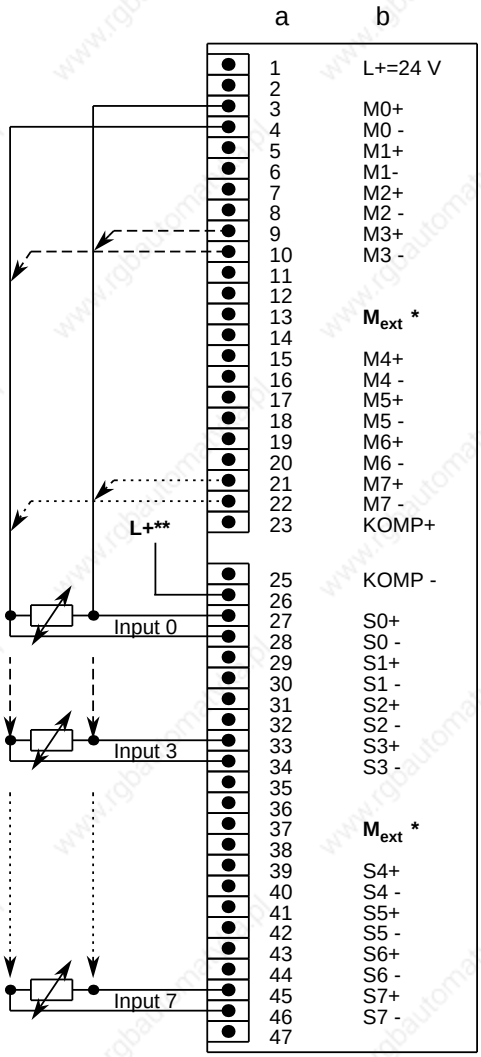
If no PT 100 is connected over input channels 4 to 7, other voltages and currents can be measured over these channels using a 498-1AA21, -1AA31, -1AA41, -1AA51, -1AA61 or -1AA71 range card (see Figure 10-14 range card 2). In this case, you must short-circuit the current outputs (S+, S-) belonging to the relevant card with a jumper. Should you fail to do so, the error bit would be set for the relevant channel and the value "0" decoded (see Figure 10-14 range card 4).

If you use a -1AA21, -1AA31 or -1AA61 range card for a channel group, no wirebreak signal may be enabled for that channel group.

A correction of 100 ohms (100 ohms=0°C) must be made via the control program by specifying the appropriate upper and lower limiting values in FB250 (see Section 10-9).

**Figure 10-14. Connecting Resistance Thermometers (PT 100s) to a 465 Analog Input Module**

The following figure shows the pin assignments of the 465-7LA13 module for resistance thermometers.




6ES5 465-7LA13

a=Pin No.
b=Assignment

* Connection to the central grounding point of the controller
** Required only for disconnecting the test current when the wirebreak signal is not activated

**Figure 10-15. Pin Assignments for Analog Input Module 465**

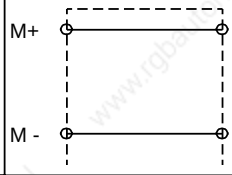
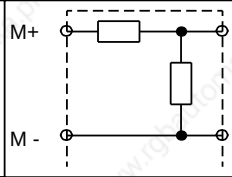
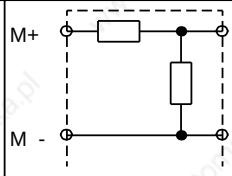
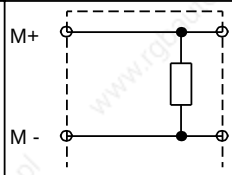
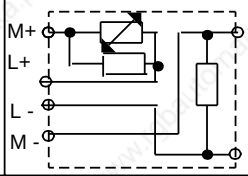
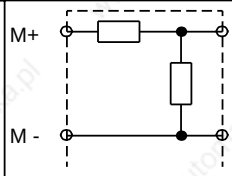
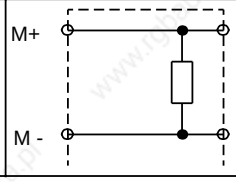
**Connecting Transducers**

Transducers are connected as in the case of the 460 module (see Section 10.2.1).

## 10.4.2 Starting Up the 465-7LA13 Analog Input Module

Voltage dividers and shunts can be plugged in as range cards (see Table 10-4). They match the process signals to the input level of the module.
In this way, various measuring ranges can be set.

**Table 10-4. Range Cards**

| Range card 6ES5 498- | Circuitry (4 times each) | Function 500 mV/mA/PT100 | Function 50 mV |
|---|---|---|---|
| - 1AA11 |  | ±500 mV;<br>PT 100 | ±50 mV |
| - 1AA21 |  | ±1 V | ±100 mV * |
| - 1AA31 |  | ±10 V | ±1 V * |
| - 1AA41 |  | ±20 mA | ±2 mA * |
| - 1AA51** |  | +4 to+20 mA<br>two-wire<br>transducer | |
| - 1AA61 |  | ±5 V | ±500 mV * |
| - 1AA71 |  | +4 to+20 mA<br>four-wire<br>transducer | |

* Possible measuring range for "50 mV" setting, but with higher incidence of error

** When a -1AA51 range card is used, there is no longer any galvanic isolation between analog inputs and L+!

**Note**

Unused inputs must be terminated with a voltage divider or shunt card. When using a through-connection card 1AA11, you must insert jumpers in the front connector.

Function select switches for setting various functions are located on the back of the 465 module. For this purpose, the switches must be set to the positions shown (see Table 10-5).

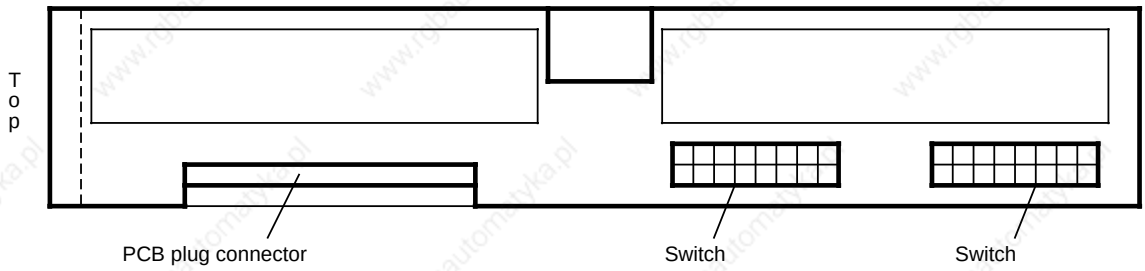
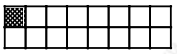
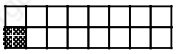
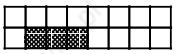
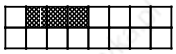
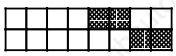
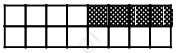
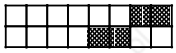
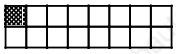
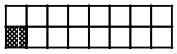
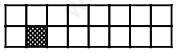
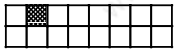
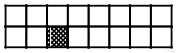
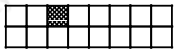
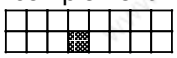
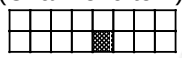
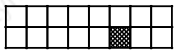
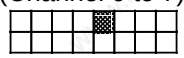
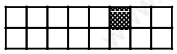
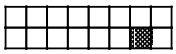
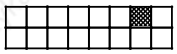



**Figure 10-16. Position of the Function Select Switches of the 465-7LA13 Analog Input Module (Rear of the Module)**

**Table 10-5. Setting Functions on the 6ES5 465-7LA13 Module**

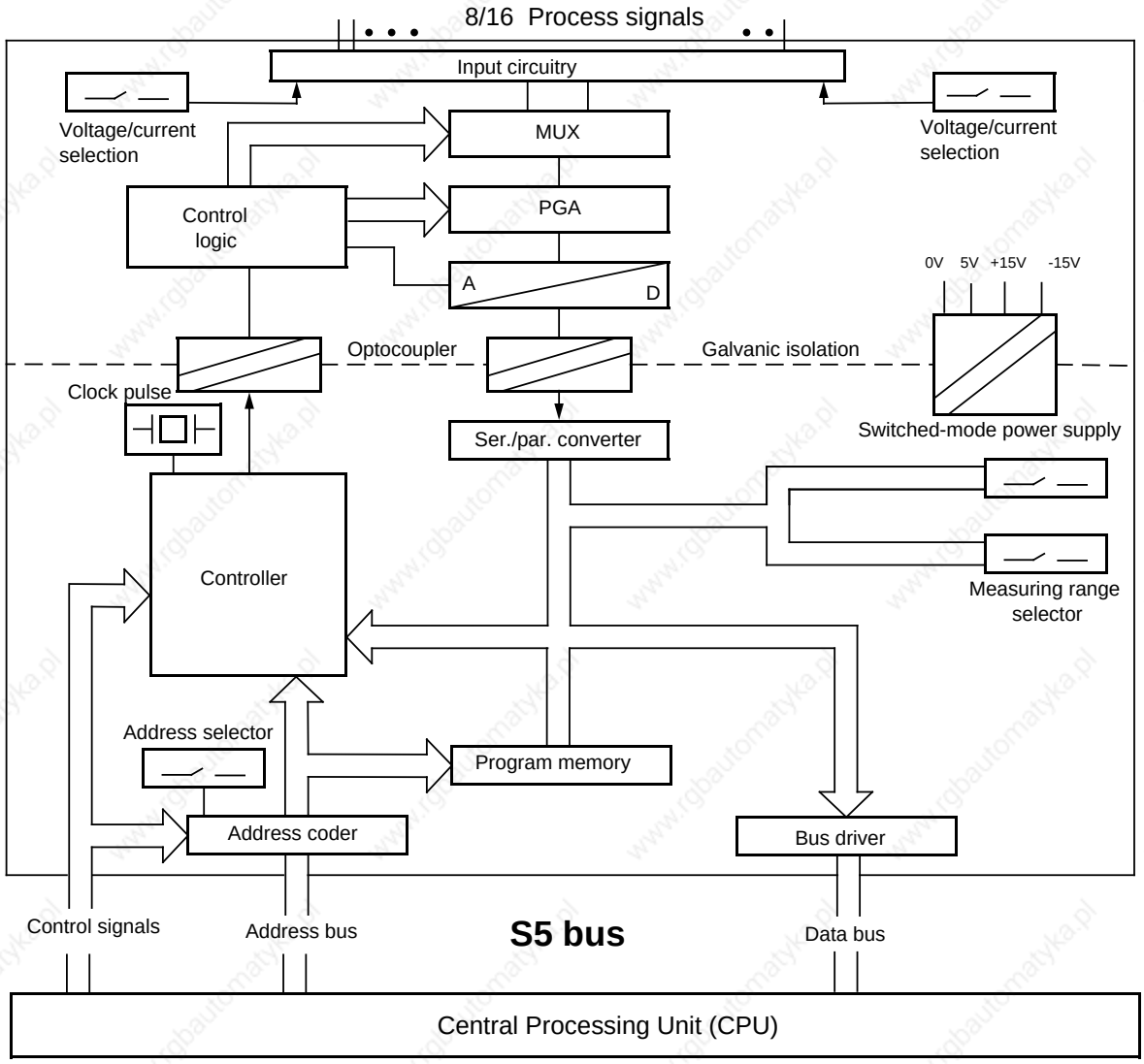
| Function | Setting on Switch | | Setting on Switch | |
|---|---|---|---|---|
| Reference junction compensation | | | Yes  | No  |
| Measuring range* (nominal value) | | | 50mV  | 500mV  |
| Measure with resistance therm., 4-wire/8-channel** | | |  | |
| Measure current or voltage | | | 8 channels  | 16 channels  |
| Sampling | Cyclic  | Selective  | | |
| System frequency | 50 Hz  | 60 Hz  | | |
| Channel operation | 8 channels  | 16 channels  | | |
| Analog value representation | Abs. value and sign | Two's complement  | | |
| Wirebreak signal for 8 channels (16 channels) | Channel 0 to 3 (Channel 0 to 7)  | Channel 4 to 7 (Channel 8 to 15)  | | |
| No wirebreak signal | Channel 0 to 3 (Channel 0 to 7)  | Channel 4 to 7 (Channel 8 to 15)  | | |
| Monitor S+line to the PT 100 resistance therm. for wirebreak | ...mV/...mA  | PT 100  | | |

* Setting for PT 100: Measuring Range 500 mV

** Additional setting for PT 100: Reference junction compensation: No

## 10.5 466-3LA11 Analog Input Module

Figure 10-17 shows the block diagram of the 466-3LA11 module.




**PGA**=Programmable amplifier

**Figure 10-17. Block Diagram of the 466-3LA11 Analog Input Module**

### Note

Please note that the 466 module has very fast processing times. Since it is very fast, it is more suitable for closed-loop control tasks than for the connection of thermocouples and resistance thermometers.

## 10.5.1 Connecting Transducers to the 466-3LA11 Analog Input Module

The pin assignments of the 466-3LA11 analog input module depend on the type of measurement (common-reference measurement or differential measurement).

**Common-reference Measurement**

| | |
|---|---|
| 1 | |
| 2 | M0+ |
| 3 | M0- |
| 4 | M8- |
| 5 | M8+ |
| 6 | |
| 7 | M1+ |
| 8 | M1- |
| 9 | M9- |
| 10 | M9+ |
| 11 | |
| 12 | M2+ |
| 13 | M2- |
| 14 | M10- |
| 15 | M10+ |
| 16 | |
| 17 | M3+ |
| 18 | M3- |
| 19 | M11- |
| 20 | M11+ |
| 21 | |
| 22 | |
| 23 | |
| 24 | M4+ |
| 25 | M4- |
| 26 | M12- |
| 27 | M12+ |
| 28 | |
| 29 | M5+ |
| 30 | M5- |
| 31 | M13- |
| 32 | M13+ |
| 33 | |
| 34 | M6+ |
| 35 | M6- |
| 36 | M14- |
| 37 | M14+ |
| 38 | |
| 39 | M7+ |
| 40 | M7- |
| 41 | M15- |
| 42 | M15+ |
| 43 | |

In the case of common-reference measurement, all signal lines have a common reference point. The reference point is produced by running all M - inputs to one point (see Figure 10-18).
Since this type of measurement is susceptible to noise, the signal sources should be located close to the 466 analog input module.

There are 16 channels available; unused channels must be short-circuited (jumper between M+ and M -).

**Labelling and Grouping of Channels**
The channels are labelled as follows on the module:

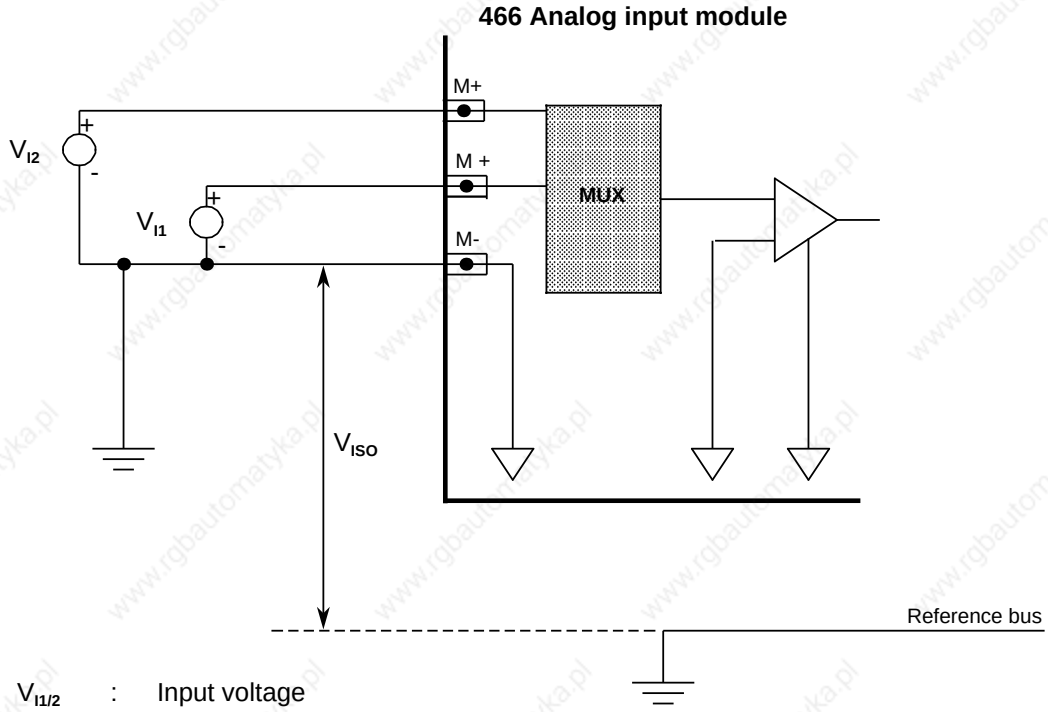
| | |
|---|---|
| Channel 0: | M0+ |
| | M0 - |
| Channel 1: | M1+ |
| | M1 - |
| : | : |
| Channel 15: | M15+ |
| | M15 - |

The channels are arranged in groups of four, for which separate measuring ranges can be set:

| | |
|---|---|
| Channel group I: | Channel 0 to 3 |
| Channel group II: | Channel 4 to 7 |
| Channel group III: | Channel 8 to 11 |
| Channel group IV: | Channel 12 to 15 |

**Figure 10-18. Pin Assignments of the 466 Analog Input Module in the Case of Common-Reference Measurement**

Figure 10-19 shows the connection of transducers to the 466 analog input module. All "M-" connection points are linked to each other internally on the module (this applies ***only*** to common-reference measurement!).




$V_{I1/2}$ : Input voltage
$V_{ISO}$ : Isolating voltage

: Equipotential: this potential is determined by the sensor reference potential (external reference potential)

**Figure 10-19. Connecting Transducers to the 466 Analog Input Module (Common-Reference Measurement)**

**Note**

See Section 3.5 for information on shielding analog signal lines!

**Differential Measurement**

| | |
|---|---|
| 1 | |
| 2 | M0+ |
| 3 | M ext |
| 4 | M ext |
| 5 | M0- |
| 6 | |
| 7 | M1+ |
| 8 | M ext |
| 9 | M ext |
| 10 | M1- |
| 11 | |
| 12 | M2+ |
| 13 | M ext |
| 14 | M ext |
| 15 | M2- |
| 16 | |
| 17 | M3+ |
| 18 | M ext |
| 19 | M ext |
| 20 | M3- |
| 21 | |
| 22 | |
| 23 | |
| 24 | M4+ |
| 25 | M ext |
| 26 | M ext |
| 27 | M4- |
| 28 | |
| 29 | M5+ |
| 30 | M ext |
| 31 | M ext |
| 32 | M5- |
| 33 | |
| 34 | M6+ |
| 35 | M ext |
| 36 | M ext |
| 37 | M6- |
| 38 | |
| 39 | M7+ |
| 40 | M ext |
| 41 | M ext |
| 42 | M7- |
| 43 | |

Differential measurement is a method of measuring which compensates for noise on the line.
Each signal line is assigned its own signal reference line. By measuring the difference between the signal line and the signal reference line, noise on both lines is compensated for.
Unused channels also must be short-circuited when using this method of measuring (jumper between M+and M -).

Differential measurement is required in the following cases:

- When the sensors are connected to different supplies
- When different signal sources are physically separate
- When signals must be captured with high accuracy
- When a high level of noise is expected.

**Labelling and Grouping of Channels**

The channels are labelled as follows on the module:

| | |
|---|---|
| | M0 - |
| Channel 1: | M1+ |
| | M1 - |
| : | : |
| Channel 7: | M7+ |
| | M7 - |

The channels are arranged in groups of four, for which separate measuring ranges can be set:

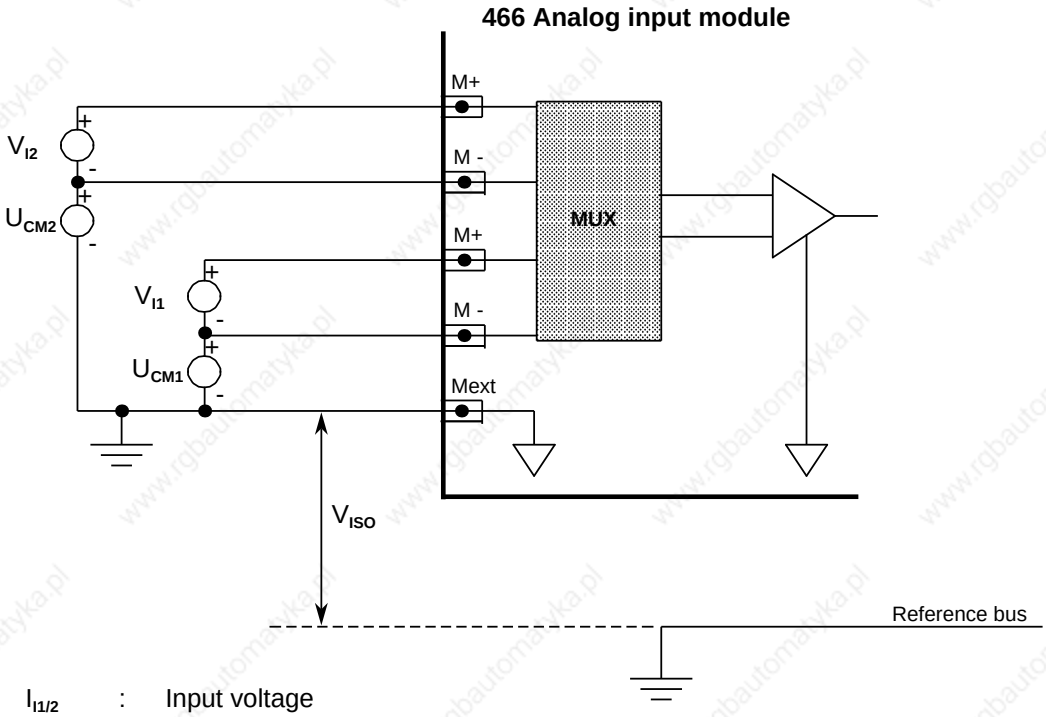
| | |
|---|---|
| Channel group I: | Group 0 to 3 |
| Channel group II: | Group 4 to 7 |

**Figure 10-20. Pin Assignments of the 466 Analog Input Module in the Case of Differential Measurement**

Figure 10-21 shows the connection of transducers to the 466 analog input module.

When connecting transducers, you must take account of the following conditions:

$V_I+V_{CM} < 12$ V (i.e. the sum of the voltage measuring set and the common mode must be less than 12 V; current measuring ranges correspond to a voltage of 2.5 V)




$I_{I1/2}$ : Input voltage
$I_{CM1/2}$ : Common mode
$I_{ISO}$ : Isolating voltage
⏚ : Equipotential: this potential is determined by the sensor reference potential (external reference potential)

**Figure 10-21. Connecting Transducers to the 466 Analog Input Module (Differential Measurement)**

**Note**

See Section 3.5 for information on shielding analog signal lines!

## 10.5.2 Start-Up of the 466-3LA11 Analog Input Module

The operating mode of the 466 analog input module is set exclusively via switches on the the printed circuit board. Figure 10-22 shows the labelling and locations of the switches on the PCB.

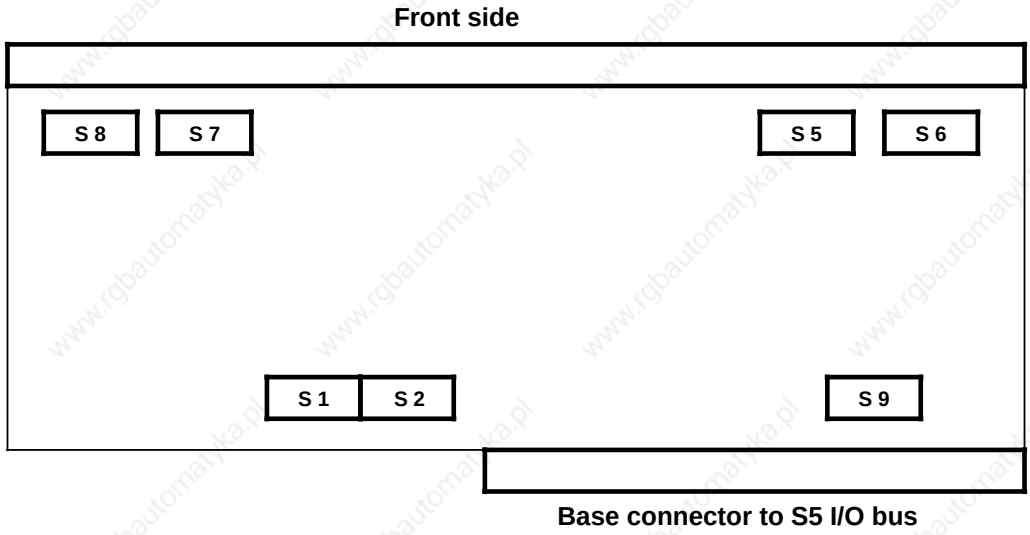



**Figure 10-22. Locations of the Mode Selectors on the 466-3LA11 Analog Input Module**

### Note

An adapter casing (e.g. 6ES5 491-0LB12) is required for using the 466 analog input module in the S5-115U.
You also require a 43-pin front connector K;

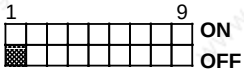
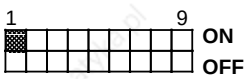
- 6XX3 068 for crimp connections
  or
- 6XX3 081 for screw connections.

**Setting the Type of Measurement**

**Common-reference Measurement/Differential Measurement**

Set switch **S 9** to the type of measurement (common-reference or differential). The switch positions refer to the module as represented in Figure 10-22:

**Table 10-6. Setting the Type of Measurement (Common-Reference or Differential)**

| Type of Measurement | Switch Position S 9 |
|---|---|
| Common-reference measurement |   |
| Differential measurement |   |

**Current/Voltage Measurement for Individual Channel Groups**

If you have set ***differential measurement*** at Switch ***S 9***, there are two channel groups available to you, each with four channels. You can configure each channel group separately for current or voltage measurement. For this purpose, you must set the switches **S 5**, **S 6**, **S 7** and **S 8** (see Table 10-7 and 10-8). Switches S 5 and S 7 permit three settings (Left, Middle, Right); switches S 6 and S 8 permit two settings (Left, Right). The switch positions refer to the module as represented in Figure 10-22:

**Table 10-7. Setting Current/Voltage Measurement for Channel Group I**

| Channel Group I (Channel 0 to 3) | Switch S 5 | Switch S 6 |
|---|---|---|
| Current | | |
| Voltage | | |

**Table 10-8. Setting Current/Voltage Measurement for Channel Group II**

| Channel Group II (Channel 4 to 7) | Switch S 7 | Switch S 8 |
|---|---|---|
| Current | | |
| Voltage | | |

If you have set ***common-reference measurement*** at Switch **S 9**, there are four channel groups available to you, each with four channels. You can configure each channel group separately for current or voltage measurement. For this purpose, you must set the switches **S 5, S 6, S 7** and **S 8** (see Table 10-9 to 10-12). Switches S 5 and S 7 permit three settings (Left, Middle, Right); switches S 6 and S 8 permit two settings (Left, Right). The switch positions refer to the module as represented in Figure 10-22:

**Table 10-9. Setting Current/Voltage Measurement for Channel Group I**

| Channel Group I (Channel 0 to 3) | Switch S 5 |
|---|---|
| Current | |
| Voltage | |

**Table 10-10. Setting Current/Voltage Measurement for Channel Group II**

| Channel Group II (Channel 4 to 7) | Switch S 7 |
|---|---|
| Current | |
| Voltage | |

**Table 10-11. Setting Current/Voltage Measurement for Channel Group III**

| Channel Group III (Channel 8 to 11) | Switch S 6 |
|---|---|
| Current | |
| Voltage | |

**Table 10-12. Setting Current/Voltage Measurement for Channel Group IV**

| Channel Group IV (Channel 12 to 15) | Switch S 8 |
|---|---|
| Current | |
| Voltage | |

### Setting the Measuring Range

The 466 analog input module has 12 measuring ranges. One measuring range can be selected for each channel group (i.e. for four inputs each), independently of the other channel groups.
Set the measuring ranges with switches S 1 and S 2. See Figure 10-23 for the assignment of switches to channel group.

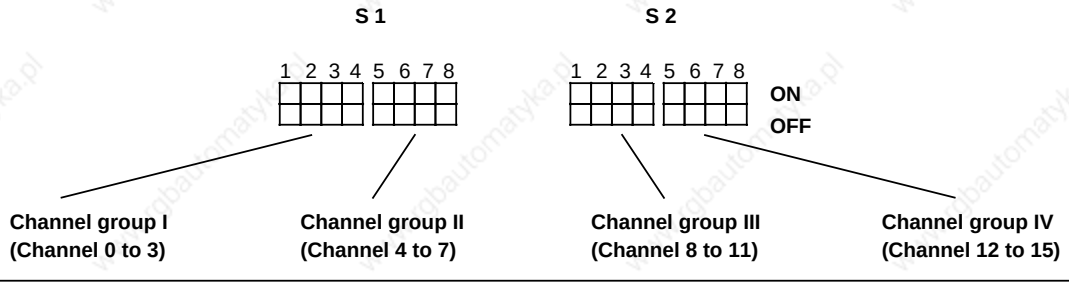



**Figure 10-23. Assignment of Switches S 1/S 2 to Channel Group**

The same measuring range coding applies to all channel groups. For this reason, the following table (see Table 10-13) contains only the measuring range setting for one channel group. The switch positions refer to the module as represented in Figure 10-22.
Please note that the type of measurement (current/voltage) must be set additionally with switches S 5 to S 8!

**Table 10-13. Setting the Measuring Range for One Channel Group (4 Channels per Group)**

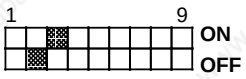
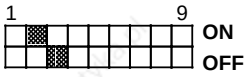
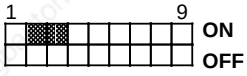
| Measuring Range | Switch Positions |
|---|---|
| 0 - 20 mA | |
| 0 - 1.25 V | |
| 0 - 2.5 V | |
| 0 - 5 V | |
| 0 - 10 V | |
| ±20 mA | |
| ±1.25 V | |
| ±2.5 V | |
| ±5 V | |
| ±10 V | |
| 4 - 20 mA | |
| 1 - 5 V | |

**Setting the Data Format**

The data format must be set with switch S 9:

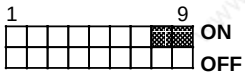
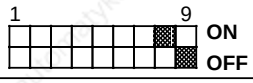
- Two's complement - 12-bit two's complement representation (range: 0 to 4095 units unipolar, or - 2048 to+2047 units bipolar)
- Number with sign - 11-bit number and 1-bit sign (range: 0 to 4095 units unipolar, or - 2048 to+2047 units bipolar)
- Binary - 12-bit binary number (range 0 to 4095 both for unipolar and bipolar variables)

**Table 10-14. Setting the Data Format**

| Data Format | Switch Position S 9 |
|---|---|
| Two's complement |   |
| Number with sign |   |
| Binary |   |

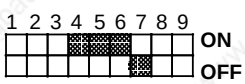
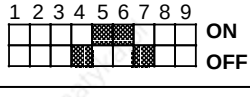
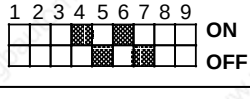
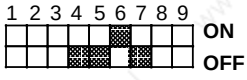
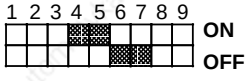
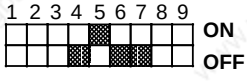
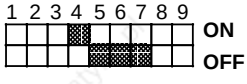
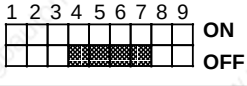
### Setting the Connection Type and the Module Starting Address

**Table 10-15. Setting the Connection Type**

| 466-3LA11 Module | Switch Position S 9 |
|---|---|
| When operating in CC or EU over distributed connections with IM 304/314, 307/317, 308/318-3 |   |
| When operating in distributed EU 701-2/3 with AS 301/310 |   |

See Table 10-16 for the precise setting of the module starting addresses.

**Table 10-16. Setting the Module Starting Addresses**

| Module Starting Address (P area) | Switch Position S 9 |
|---|---|
| 128 ($F080_H$) |   |
| 144* ($F090_H$) |   |
| 160 ($F0A0_H$) |   |
| 176* ($F0B0_H$) |   |
| 192 ($F0C0_H$) |   |
| 208* ($F0D0_H$) |   |
| 224 ($F0E0_H$) |   |
| 240* ($F0F0_H$) |   |

* Can only be set in the case of differential measurement

## 10.6 Representation of the Digital Input Value

The analog value has the same representation in the three analog input modules.
However, there are differences in the case of analog value evaluation where the individual analog input modules are concerned, especially bits 0 to 2 (see Figure 10-24).

After an analog signal is converted, the digital result is stored in the module's RAM. Figure 10-24 explains the individual bits of the two bytes.

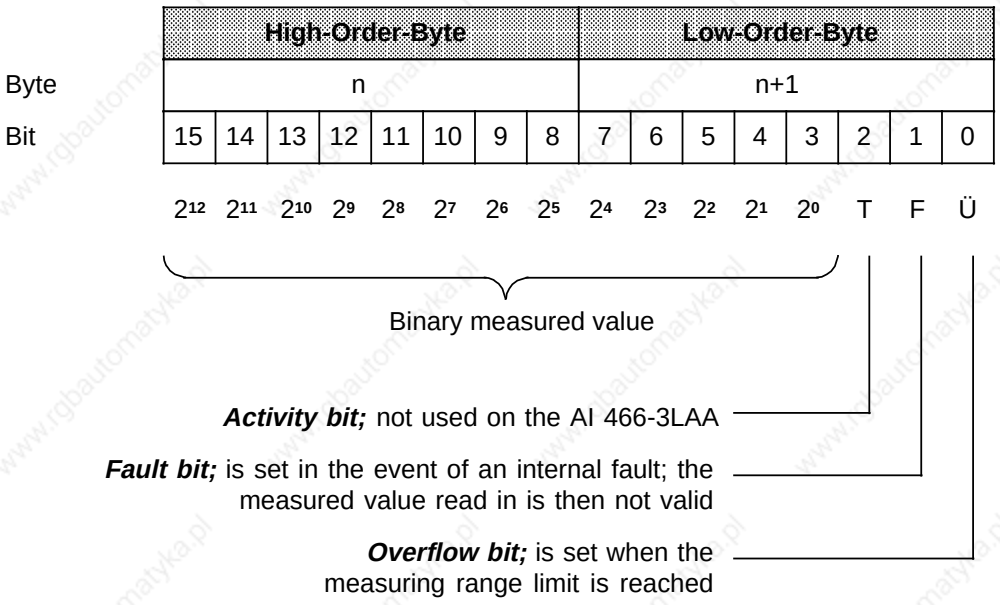



**Figure 10-24. Representation of the Digitized Measured Value**

Bits 0 to 2 are irrelevant for the measured value. They provide information on the measured value representation. Table 10-17 describes these bits in detail.

**Table 10-17. Meaning of Bits 0 to 2 for Analog Input Modules**

| Bit | Meaning | Signal State | Meaning of the Signal State |
|---|---|---|---|
| OV | Overflow bit | 1 | Range exceeded* |
| F | Fault bit | 1 | Wire break |
| A | Activity bit | 0 | Cyclic scan or "Not active" (for single scan) |
| | | 1 | Coding procedure for single scan not yet terminated |

* An overflow at one measuring point has no effect on the overflow bits of the other channels, i.e. the values on the other channels are correct and may be evaluated.

**Special Features of the 466 Module**

- Bit 15 ($2^{12}$) indicates the sign in the case of bipolar measured value representation (two's complement and number with sign).
- Bit 14 ($2^{11}$) is not used in the case of bipolar measured value representation (no overrange!).
- The 466 module has no overrange.
- Selective sampling is not possible on the 466 module (activity bit is not set).

**Types of Representation for the 460 and 465 Analog Input Modules**

The way in which the analog value is represented depends on the type of module (see Tables 10-18 to 10-24).

**Table 10-18. Representation of Digitized Measured Values of the 460 and 465 AI (*Two's Complement*; Measuring Range±50 mV,±500 mV,±1000 mV)**

| Meas. Value in mV (±50) | Meas. Value in mV (±500) | Meas. Value in mV (±1000) | Units | Digitized Measured Value | | | | | | | | | | | | | T F OV | | | Range |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 100.0 | 1000.0 | 2000.0 | 4095+0V | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | |
| 99.976 | 999.76 | 1999.52 | 4095 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| 50.024 | 500.24 | 1000.48 | 2049 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| 50.000 | 500.00 | 1000.00 | 2048 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 49.976 | 499.76 | 999.52 | 2047 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| 25.000 | 250.00 | 500.00 | 1024 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 24.976 | 249.76 | 499.52 | 1023 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| 0.024 | 0.24 | 0.48 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| 0.000 | 0.00 | 0.00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| -0.024 | -0.24 | -0.48 | -1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| -24.976 | -249.76 | -499.52 | -1023 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| -25.000 | -250.00 | -500.00 | -1024 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| -49.976 | -499.76 | -999.52 | -2047 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| -50.000 | -500.00 | -1000.00 | -2048 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| -50.024 | -500.24 | -1000.48 | -2049 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| -99.976 | -999.76 | -1999.52 | -4095 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| -100.0 | -1000.0 | -2000.0 | -4095+0V | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | |

Overflow
Overrange
Nominal range

**Table 10-19. Representation of Digitized Measured Values of the AI 460 and 465 (*Two's Complement*; Measuring Range±5 V,±10 V,±20 mA)**

| Meas. Value in V (±5) | Meas. Value in V (±10) | Meas. Value in mA (±20) | Units | Digitized Measured Value | | | | | | | | | | | | | TF OV | | | Range |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 10.00 | 20.00 | 40.00 | 4095+OV | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | |
| 9.9976 | 19.9952 | 39.9902 | 4095 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| 5.0024 | 10.0048 | 20.0098 | 2049 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| **5.0000** | **10.0000** | **20.0000** | **2048** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **4.9976** | **9.9952** | **19.9902** | **2047** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| **2.5000** | **5.0000** | **10.0000** | **1024** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **2.4976** | **4.9952** | **9.9902** | **1023** | **0** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| **0.0024** | **0.0048** | **0.0098** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | |
| **0.0000** | **0.0000** | **0.0000** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **-0.0024** | **-0.0048** | **-0.0098** | **-1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| **-2.4976** | **-4.9952** | **-9.9902** | **-1023** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | |
| **-2.5000** | **-5.0000** | **-10.0000** | **-1024** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| **-4.9976** | **-9.9952** | **-19.9902** | **-2047** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | |
| **-5.0000** | **-10.0000** | **-20.0000** | **-2048** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| -5.0024 | -10.0048 | -20.0098 | -2049 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| -9.9976 | -19.9952 | -39.9902 | -4095 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| -10.00 | -20.00 | -40.00 | -4095+OV | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | |

Overflow
Overrange
Nominal range

**Table 10-20. Representation of Digitized Measured Values of the 460 and 465 AI (Number and Sign; Mesuring Range±50 mV,±500 mV,±1000 mV)**

| Meas. Value in mV (±50) | Meas. Value in mV (±500) | Meas. Value in mV (±1000) | Units | Digitized Measured Value | | | | | | | | | | | | | | TF OV | | | Range |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | | 2 | 1 | 0 | |
| 100.0 | 1000.0 | 2000.0 | 4095+OV | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | 0 | 0 | 1 | |
| 99.976 | 999.76 | 1999.52 | 4095 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | | |
| 50.024 | 500.24 | 1000.48 | 2049 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | | 0 | 0 | 0 | |
| **50.000** | **500.00** | **1000.00** | **2048** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | | **0** | **0** | **0** | |
| **49.976** | **499.76** | **999.52** | **2047** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | | **0** | **0** | **0** | |
| **:** | **:** | **:** | **:** | | | | | | | | **:** | | | | | | | | | | |
| **25.000** | **250.00** | **500.00** | **1024** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | | **0** | **0** | **0** | |
| **24.976** | **249.76** | **499.52** | **1023** | **0** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | | **0** | **0** | **0** | |
| **:** | **:** | **:** | **:** | | | | | | | | **:** | | | | | | | | | | |
| **0.0024** | **0.24** | **0.48** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | | **0** | **0** | **0** | |
| **0.0000** | **0.00** | **0.00** | **+0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | | **0** | **0** | **0** | |
| **0.0000** | **0.00** | **0.00** | **-0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | | **0** | **0** | **0** | |
| **-0.0024** | **-0.24** | **-0.48** | **-1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | | **0** | **0** | **0** | |
| **:** | **:** | **:** | **:** | | | | | | | | **:** | | | | | | | | | | |
| **-24.976** | **-249.76** | **-499.52** | **-1023** | **1** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | | **0** | **0** | **0** | |
| **-25.000** | **-250.00** | **-500.00** | **-1024** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | | **0** | **0** | **0** | |
| **:** | **:** | **:** | **:** | | | | | | | | **:** | | | | | | | | | | |
| **-49.976** | **-499.76** | **-999.52** | **-2047** | **1** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | | **0** | **0** | **0** | |
| **-50.000** | **-500.00** | **-1000.00** | **-2048** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | | **0** | **0** | **0** | |
| -50.024 | -500.24 | -1000.48 | -2049 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | | |
| -99.976 | -999.76 | -1999.52 | -4095 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | 0 | 0 | 0 | |
| -100.0 | -1000.0 | -2000.0 | -4095+OV | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | 0 | 0 | 1 | |

Overflow
Overrange
Nominal range

## Note

Bit 7 in the high-order byte is the sign (S).
If S is 0, the value is positive. If S is 1, the value is negative.

**Table 10-21. Representation of Digitized Measured Values of the 460 and 465 AI (*Number and Sign*; Measuring Range±5 V,±10 V,±20 mA)**

| Meas. Value in V (±5) | Meas. Value in V (±10) | Meas. Value in mA (±20) | Units | Digitized Measured Value | | | | | | | | | | | | | T F OV | | | Range |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 10.00 | 20.00 | 40.00 | 4095+OV | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | |
| 9.9976 | 19.9952 | 39.9902 | 4095 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| 5.0024 | 10.0048 | 20.0098 | 2049 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| **5.0000** | **10.0000** | **20.0000** | **2048** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **4.9976** | **9.9952** | **19.9902** | **2047** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | |
| **:** | **:** | **:** | **:** | | | | | | | | **:** | | | | | | | | | |
| **2.5000** | **5.0000** | **10.0000** | **1024** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **2.4976** | **4.9952** | **9.9902** | **1023** | **0** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | |
| **:** | **:** | **:** | **:** | | | | | | | | **:** | | | | | | | | | |
| **0.0024** | **0.0048** | **0.0098** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | |
| **0.0000** | **0.0000** | **0.0000** | **+0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **0.0000** | **0.0000** | **0.0000** | **-0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **-0.0024** | **-0.0048** | **-0.0098** | **-1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | |
| **:** | **:** | **:** | **:** | | | | | | | | **:** | | | | | | | | | |
| **-2.4976** | **-4.9952** | **-9.9902** | **-1023** | **1** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | |
| **-2.5000** | **-5.0000** | **-10.0000** | **-1024** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **:** | **:** | **:** | **:** | | | | | | | | **:** | | | | | | | | | |
| **-4.9976** | **-9.9952** | **-19.9902** | **-2047** | **1** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | |
| **-5.0000** | **-10.0000** | **-20.0000** | **-2048** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| -5.0024 | -10.0048 | -20.0098 | -2049 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| : | : | : | : | | | | | | | | : | | | | | | | | | |
| -9.9976 | -19.9952 | -39.9902 | -4095 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | |
| -10.00 | -20.00 | -40.00 | -4095+OV | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | |

Overflow
Overrange
Nominal range

Set the measuring range of the module to 500 mV and plug in a 6ES5 498-1AA 71 module.
The measuring range 4 to 20 mA is resolved into 2048 units from 512 to 2560. For representation in the range 0 to 2048, 512 units must be subtracted at the software level.

**Table 10-22. Representation of Digitized Measured Values of the 460 and 465 AI (Current Measuring Range 4 to 20 mA)**

| $I_I$ in mA | Units | $V_I$ in mV | Digitized Measured Value | | | | | | | | | | | | | T | F | ov | Range |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 32.796 | 4096+OV | 1024 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | Overflow |
| 31.992 | 4095 | 1023.76 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | Over- |
| 24.0 | 3072 | 750.0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | range* |
| 23.992 | 3071 | 749.76 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | |
| 20.008 | 2561 | 625.24 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | |
| **20.0** | **2560** | **625.0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **Nominal** |
| **16.0** | **2048** | **500.0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **range** |
| **4.0** | **512** | **125.0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| 3.992 | 511 | 124.76 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | Below |
| 3.0 | 384 | 93.75 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | nominal |
| 2.992 | 383 | 93.5 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | range |
| 0.0 | 0 | 0.0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Wirebreak |

* Short-circuit of the two-wire transducer

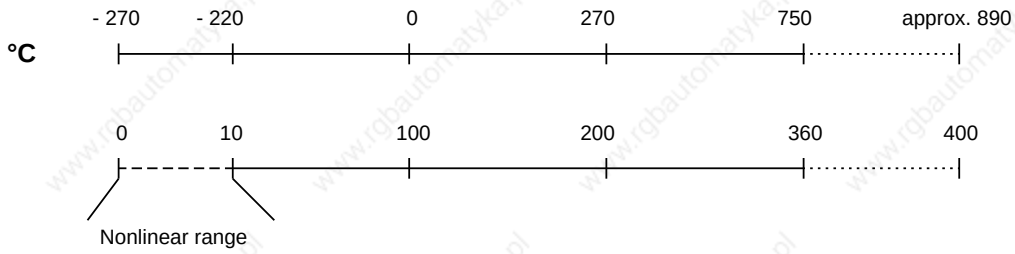
## Note

The 31.25 shunt resistor integrated in the 498-1AA71 suppresses the wire break signal (the F bit is not set). You can thus detect a wire break only by comparing the measured value with a lower limiting value in the user program. A measured value lower than, for example, 1 mA (=128 units) would then be interpreted as a wire break.

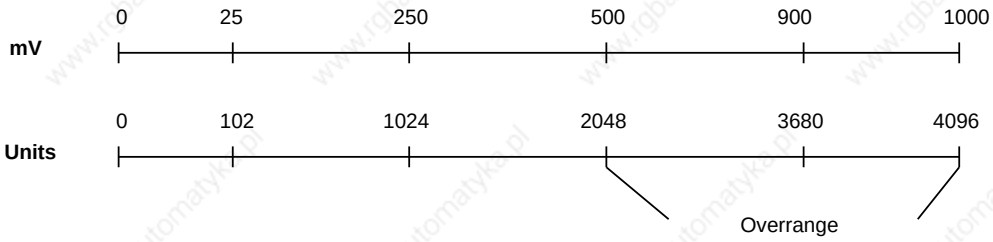
**Table 10-23. Representation of Digitized Measured Values of the 460 and 465 AI for Resistance-Type Sensors**

| Sensor Resistance ( ) | Units | Digitized Measured Value | | | | | | | | | | | | | T | F | OV | Range |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 400.0 | 4095 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | Overflow |
| 399.90 | 4095 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | Over- |
| 200.098 | 2049 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | range |
| **200.00** | **2048** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **199.90** | **2047** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | **Nominal** |
| **100.0** | **1024** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **range** |
| **99.90** | **1023** | **0** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | |
| **0.098** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | |
| **0.0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |

The resolution in the case of the PT 100 is approximately 1/3 °C 10 units correspond to approximately 1 .
You can use the assignment in Figure 10-25 for the PT 100 resistance sensor.
Linearization of the digital input values is not carried out via the modules. You can linearize the input values only via the relevant software solution.




**U=R · I=R · 2.5 mA (constant current)**




**Resolution: 10 units=1**
**270 °C : 1024 units=0.3 °C/unit**

**Figure 10-25. PT 100 on SIMATIC Analog Input Modules**

**Representation of Measured Value for the New PT100 Climatic Measuring Range of the 460-7LA13 AI**

**Table 10.24 Representation of Digitized Measured Values for the PT100 Climatic Measuring Range of the 460-7LA13 AI**

| Units | PT100/ ohm | °C | mV at PT100 | mV compensated | Digitized Measured Value 15 14 13 12 11 10 9 8 7 6 5 4 3 | T 2 | F 1 | OV 0 | Range |
|---|---|---|---|---|---|---|---|---|---|
| >4095 | 140 | | 350 | 100 | 0 1 1 1 1 1 1 1 1 1 1 1 1 | 0 | 0 | 1 | Overflow |
| +4095 | 139.99 | 103.74 | 349.976 | 99.976 | 0 1 1 1 1 1 1 1 1 1 1 1 1 | 0 | 0 | 0 | Overrange |
| +2049 | 120.01 | 51.61 | 300.024 | 50.024 | 0 1 0 0 0 0 0 0 0 0 0 0 1 | 0 | 0 | 0 | |
| +2048 | 120 | 51.580 | 300 | 50 | 0 1 0 0 0 0 0 0 0 0 0 0 0 | 0 | 0 | 0 | Nominal |
| +1 | 100.01 | 0.026 | 250.024 | 0.024 | 0 0 0 0 0 0 0 0 0 0 0 0 1 | 0 | 0 | 0 | range |
| 0 | 100 | 0 | 250 | 0 | 0 0 0 0 0 0 0 0 0 0 0 0 0 | 0 | 0 | 0 | |
| -1 | 99.99 | -0.026 | 249.976 | -0.024 | 1 1 1 1 1 1 1 1 1 1 1 1 1 | 0 | 0 | 0 | |
| -2048 | 80 | -50.780 | 200 | -50 | 1 1 0 0 0 0 0 0 0 0 0 0 0 | 0 | 0 | 0 | |
| -2049 | 79.99 | -50.81 | 199.976 | -50.024 | 1 0 1 1 1 1 1 1 1 1 1 1 1 | 0 | 0 | 0 | Overrange |
| -4095 | 60.01 | -100.60 | 150.024 | -99.976 | 1 0 0 0 0 0 0 0 0 0 0 0 1 | 0 | 0 | 0 | |
| <-4095 | 60 | | 150 | -100 | 1 0 0 0 0 0 0 0 0 0 0 0 1 | 0 | 0 | 1 | Overflow |
| Wire break | | | | | | | | | |
| -4095 | x | x | 0 | -250 | 1 0 0 0 0 0 0 0 0 0 0 0 1 | 0 | 0 | 1 | Wire break $I_{C+}/I_{C-}$ 2) |
| -4095 | x | x | 0 | -250 | 1 0 0 0 0 0 0 0 0 0 0 0 1 | 0 | 1 | 1 | Wire break transducer measuring circuit 1) |

T: Active F: Error bit OV: Overflow X: Any

1) Only with wire break detection activated
Error bit = 1 only in case of faulty channel;
In case of transducer break overflow bit = 1 for all channels

2) Through the PT100 series connection, this bit combination is set for all channels in the case of a break in the supply circuit.

**Forms of Representation for the 466 Analog Input Module**

Tables 10-25 to 10-33 give information on the representation of the digitized measured value depending on the measuring range selected.
The 466 analog input module has no overrange.

**Table 10-25. Representation of Digitized Measured Values of the 466 AI (Measuring Range 0 to 20 mA; 0 to 5 V and 0 to 10 V; unipolar)**

| Meas. Value in V (0 to 5 V) | Meas. Value in V (0 to 10 V) | Meas. Value in mA (0 to 20 mA) | Units | Digitized Measured Value* | | | | | | | | | | | | | T | F | OV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 4.9988 | 9.9976 | 19.9951 | 4095 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 4.9976 | 9.9951 | 19.9902 | 4094 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| : | : | : | : | | | | | | : | | | | | | | | | : | |
| 0.0012 | 0.0024 | 0.00488 | 0001 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0.0000 | 0.0000 | 0.00000 | 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

***** Same representation as for two's complement, number and sign and binary representation

**Table 10-26. Representation of Digitized Measured Values (*Two's Complement;* Measuring Range±5 V,±20 mA and±10 V; bipolar)**

| Meas. Value in V (±5 V) | Meas. Value in V (±10V) | Meas. Value in mA (±20mA) | Units | Digitized Measured Value | | | | | | | | | | | | | T | F | OV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 4.9976 | 9.9951 | 19.9902 | 2047 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 4.9951 | 9.9902 | 19.9804 | 2046 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| : | : | : | : | | | | | | | : | | | | | | | | : | |
| 0.0024 | 0.0049 | 0.00976 | 0001 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0.0000 | 0.0000 | 0.00000 | 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -0.0024 | -0.0049 | -0.00976 | -0001 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| : | : | : | : | | | | | | | : | | | | | | | | : | |
| -4.9976 | -9.9951 | -19.9902 | -2047 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| -5.0000 | -10.000 | -20.0000 | -2048 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Table 10-27. Representation of Digitized Measured Values (*Number and Sign*; Measuring Range±5 V,±20 mA and±10 V; bipolar)**

| Meas. Value in V (±5 V) | Meas. Value in V (±10 V) | Meas. Value in mA (±20 mA) | Units | Digitized Measured Value | | | | | | | | | | | | | T | F | OV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 4.9976 | 9.9951 | 19.9902 | 2047 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 4.9951 | 9.9902 | 19.9804 | 2046 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| : | : | : | : | | | | | | | : | | | | | | | | : | |
| 0.0024 | 0.0049 | 0.00976 | 0001 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0.0000 | 0.0000 | 0.00000 | 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -0.0024 | -0.0049 | -0.00976 | -0001 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| : | : | : | : | | | | | | | : | | | | | | | | : | |
| -4.9976 | -9.9951 | -19.9902 | -2047 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| -5.0000 | -10.000 | -20.0000 | -2048 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Table 10-28. Representation of Digitized Measured Values (*Binary*; Measuring Range±5 V,±20 mA and±10 V; bipolar)**

| Meas. Value in V (±5 V) | Meas. Value in V (±10 V) | Meas. Value in mA (±20 mA) | Units | Digitized Measured Value | | | | | | | | | | | | | T | F | OV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 4.9976 | 9.9951 | 19.9902 | 4095 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 4.9951 | 9.9902 | 19.9804 | 4094 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| : | : | : | : | | | | | | | : | | | | | | | | : | |
| 0.0024 | 0.0049 | 0.00976 | 2049 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0.0000 | 0.0000 | 0.00000 | 2048 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -0.0024 | -0.0049 | -0.00976 | 2047 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| : | : | : | : | | | | | | | : | | | | | | | | : | |
| -4.9976 | -9.9951 | -19.9902 | 0001 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| -5.0000 | -10.000 | -20.0000 | 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Table 10-29. Representation of Digitized Measured Values (Measuring Range 0 to 1.25 V, and 0 to 2.5 V; unipolar)**

| Meas. Value in V (0 to 1.25 V) | Meas. Value in V (0 to 2.5 V) | Units | Digitized Measured Value* | | | | | | | | | | | | | T | F | OV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 1.2497 | 2.4994 | 4095 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1.2494 | 2.4988 | 4094 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| : | : | : | | | | | | | : | | | | | | | | : | |
| 0.0003 | 0.0006 | 0001 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0.0000 | 0.0000 | 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

***** Same representation as for two's complement data format, number and sign and binary representation

**Table 10-30. Representation of Digitized Measured Values (*Two's Complement*; Measuring Range±1.25 V, and±2.5 V; bipolar)**

| Meas. Value in V (±1.25 V) | Meas. Value in V (±2.5 V) | Units | Digitized Measured Value | | | | | | | | | | | | | T | F | OV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 1.2494 | 2.4988 | 2047 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1.2488 | 2.4975 | 2046 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| : | : | : | | | | | | | : | | | | | | | | : | |
| 0.0006 | 0.0012 | 0001 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0.0000 | 0.0000 | 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -0.0006 | -0.0012 | -0001 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| : | : | : | | | | | | | : | | | | | | | | : | |
| -1.2494 | -2.4988 | -2047 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| -1.2500 | -2.5000 | -2048 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Table 10-31. Representation of Digitized Measured Values (*Number and Sign*; Measuring Range±1.25 V, and±2.5 V; bipolar)**

| Meas. Value in V (±1.25V) | Meas. Value in V (±2.5 V) | Units | Digitized Measured Value | | | | | | | | | | | | | T | F | OV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 1.2494 | 2.4988 | 2047 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1.2488 | 2.4975 | 2046 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| : | : | : | | | | | | | : | | | | | | | | : | |
| 0.0006 | 0.0012 | 0001 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0.0000 | 0.0000 | 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -0.0006 | -0.0012 | -0001 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| : | : | : | | | | | | | : | | | | | | | | : | |
| -1.2494 | -2.4988 | -2047 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.2500 | -2.5000 | -2048 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Table 10-32. Representation of Digitized Measured Values (*Binary*; Measuring Range±1.25 V, and±2.5 V; bipolar)**

| Meas. Value in V (±1.25V) | Meas. Value in V (±2.5 V) | Units | Digitized Measured Value | | | | | | | | | | | | | T | F | OV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 1.2494 | 2.4988 | 4095 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |
| 1.2488 | 2.4975 | 4094 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| : | : | : | | | | | | | : | | | | | | | | : | |
| 0.0006 | 0.0012 | 2049 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0.0000 | 0.0000 | 2048 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -0.0006 | -0.0012 | 2047 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| : | : | : | | | | | | | : | | | | | | | | : | |
| -1.2494 | -2.4988 | 0001 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| -1.2500 | -2.5000 | 0000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

**Table 10-33. Representation of Digitized Measured Values (Measuring Range 4 to 20 mA and 1 to 5 V)**

| Meas. Value in V (1 to 5 V) | Meas. Value in mA (4 to 20 mA) | Units | Digitized Measured Value* | | | | | | | | | | | | | | T | F | OV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | | 2 | 1 | 0 |
| 4.998 | 19.992 | 2559 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | 0 | 0 | 1 |
| 4.000 | 16.000 | 2048 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |
| 1.000 | 4.000 | 512 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |
| 0.998 | 3.992 | 511 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | 0 | 0 | 0 |
| 0.750 | 3.000 | 384 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 |
| 0.748 | 2.992 | 383 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | 0 | 0 | 0 |
| 0.000 | 0.000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 1 |

***** Same representation as for two's complement data format, number and sign and binary representation

The measuring ranges 4 to 20 mA and 1 to 5 V (see Table 10-33) are resolved to 2048 units in the interval 512 to 2560. For representation in the range 0 to 2048, 512 units must be subtracted per software.

A wirebreak signal is not provided. You can scan the measured value in the user program for a lower limit and interpret values below this limit as wirebreak.

## 10.7 Wirebreak Signal and Sampling for Analog Input Modules

**Wirebreak Signal**

Wirebreak is signalled **only** in the case of the **460** and **465** analog input modules.

If a 6ES5 498-1AA11 range card (through-connection card) is used, you can select the "Wirebreak signal" function to monitor the sensors connected to the inputs (see Tables 10-1 to 10-4). You can select wirebreak detection for 8 or 16 inputs for 16-channel operation or for 4/8 inputs for 8-channel operation.

The wirebreak signal is issued under the following conditions:
Before each input value is decoded, a constant current is applied briefly (1.6 ms) to the input terminals and the resulting voltage compared with a limiting value. If the sensor circuit or supply lead is interrupted, the voltage exceeds the limiting value and a wirebreak signal is generated (bit 1 is set in data byte 1; refer to Section 10.5.1). The ADC decodes the value "0".
When the signal at the input is measured with a digital voltmeter, the constant-current pulses may cause apparent fluctuations in the signal. When the input circuit that supplies the analog value has capacitive characteristics, the constant current falsifies the measured value.
Should these apparent fluctuations in the signal prove annoying, e.g. on startup, the test current can be deactivated on the 460 and 465 analog input modules by applying +24 V to pin 26 in the front connector and 0 V to pin 47 (L-) or to analog input module 465, pin 37 ($M_{ext}$). In addition, mode selector I must be set to "No wirebreak signal".

A wirebreak signal serves a practical purpose only in conjunction with a 6ES5 498-1AA11 through connection card. It is not possible to detect a wirebreak on the 6ES5 498-1AA41, -1AA51 or -1AA71 range cards, as the measuring inputs are terminated with low-resistance shunts. On all other range cards, a wirebreak signal results in an undefined reaction.

**Wirebreak Signal in Conjunction with Resistance Thermometers**

An interruption in the supply leads to a resistance thermometer is reported as follows:

**Table 10-34. Wirebreak Signal in Conjunction with Resistance Thermometers**

| Wirebreak on | Digitized Analog Value (460/465 Module) | Status of the Error Bit (460 Module) | Status of the Error Bit (465 Module) |
|---|---|---|---|
| M+ | 0/0 | 1 | 1 |
| M - | 0/0 | 1 | 1 |
| PT 100 (resist.-type sensor) | 0*/0 | 0* | 1 |
| S+ | 0/0 | 0 | 1 |
| S - | 0/0 | 0 | 1 |

* On the 460 analog input module, the value "0" is also decoded for the unbroken PT 100 resistors and error bit F set to 0.

The overflow bit is set separately for each channel in the case of the 460/465-7LA12 modules.

The S+lines to the resistance thermometer can be monitored for a wirebreak on the 465 -7LA12 analog input module by setting switch 7 of mode selector I to "PT 100" (PT 100 constant power supply). The error bit is also set to flag a wirebreak in this line.
Unused channels can be used to measure voltages or currents when the current sourcing outputs (S+, S -) associated with the relevant measuring channel are short-circuited with a jumper. Without this jumper, the error bit would be set for this channel and the value "0" decoded.
The S+lines are not monitored for wirebreak when mode selector II is in the "Current or voltage measurement" position. In this case, the error bit is not set when a wirebreak occurs. This switch setting should be selected when only voltages or currents are to be measured (see Figure 10-7).

The following general rule applies: When the wirebreak signal is to be issued, the measuring circuit must have a low resistance (<1 k ).

### Sampling

The **460** and **465** modules offer two methods of sampling the analog value:
- Cyclic sampling and
- Selective sampling

The **466** module implements only cyclic sampling because of its high speed.

### Cyclic Sampling

The modules's processor decodes all inputs.
However, there are differences between the individual modules.
For example, the amount of time that elapses before a measured value is updated depends on the number of input channels. The time required for decoding depends on the input value. In the case of the 460 analog input module, when $V_I$=0 V, decoding takes 40 ms; when $V_I$=nominal value, decoding takes 60 ms.

**Table 10-35. Scan Times**

| Module | 460 | 465 | | 466 | |
|---|---|---|---|---|---|
| Channels | 8 | 8 | 16 | 8 | 16 |
| Scan Time* | 480 ms | 480 ms | 960 ms | 2 ms | 4 ms |

* Nominal value applied to all inputs

In the case of the 460/465 modules, the digitized measured values are stored in the circulating buffer under the channel address (the high-order byte under address n, the low-order byte under address n+1), and can be read out from the buffer whenever required.

### Selective Sampling

Selective sampling is ***not*** possible on the **466** module.
**Double addressing** cannot be used for selective sampling, i.e. an address cannot be assigned to an analog output module and an analog input module.
In the case of the 460 and 465 modules, the initiative for decoding a measured value comes from the CPU when this function is used. The module must be accessed once with a Write command (T PW) under the relevant channel address; the data itself is of no relevance. In this way, only the measured value of the activated channel is decoded and the other channels are ignored. During decoding, an activity bit is set on the data bus (**A=1**, see also Section 10.6). The module sets the activity bit independently, i.e. if several channels are to be decoded using selective sampling, the activity bit cannot be assigned to one channel! The valid digitized measured value can be read out from two bytes once the activity bit has been reset (A=0, negative-going edge).
Repeated scanning of the activity bit loads both the bus and the CPU. This results in non-periodic measured value acquisition when different measured values are involved, and is therefore not desirable for PID control tasks.
A better method is time-controlled program scanning, in which certain program sections, for instance FB13, are automatically inserted into the program every 100 ms by a time-controlled block (OB13), thus producing a constant time base while offloading the bus and the CPU.

The associated sample program is written as follows:

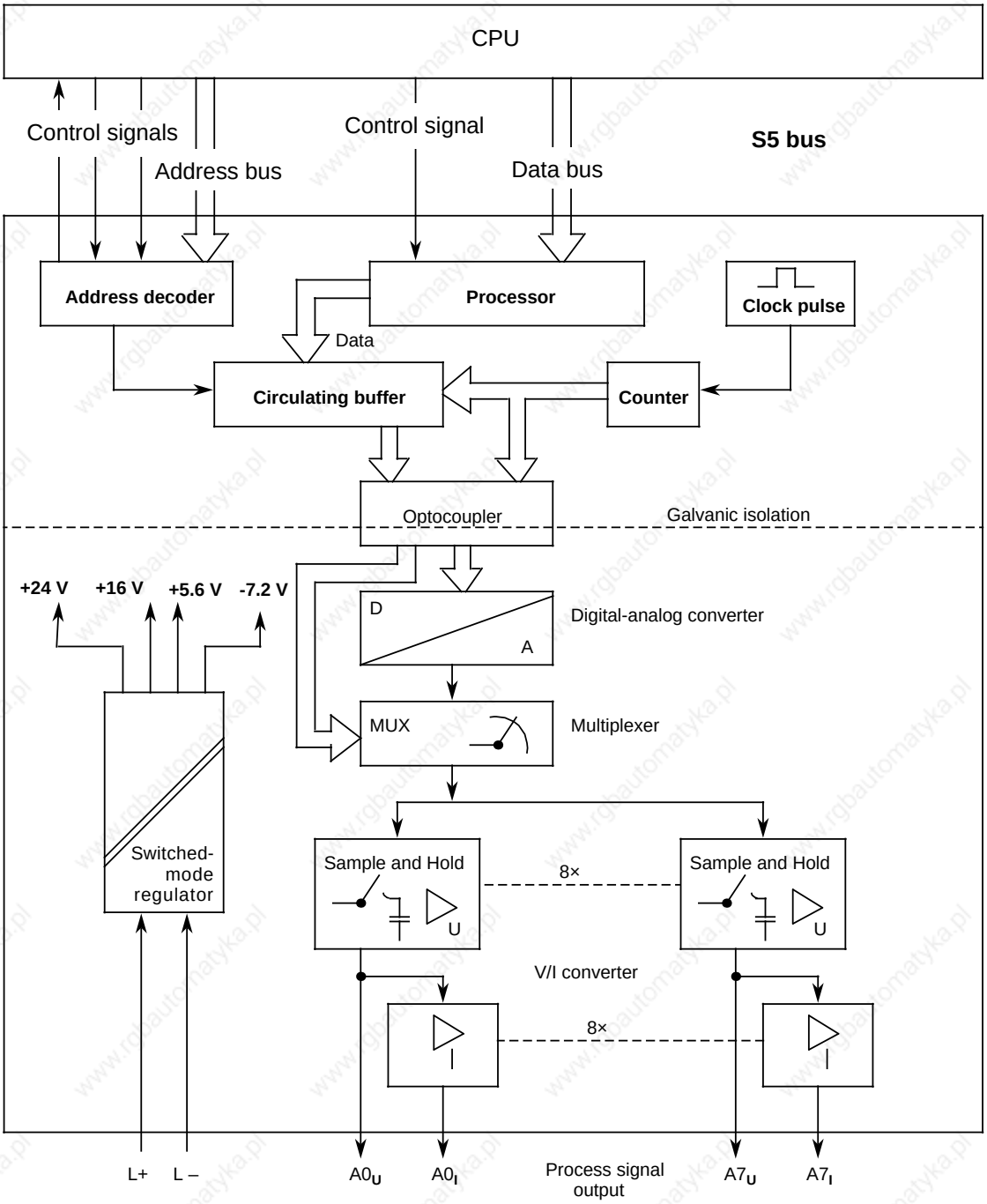
| FB13 STL | Description |
|---|---|
| NAME:SEL-SAMP | EXAMPLE FOR SELECTIVE SAMPLING |
| :L PW128 | READ ANALOG VALUE |
| :T FW128 | TRANSFER TO AUXILIARY FLAG |
| :A F 129.2 | SCAN ACTIVITY BIT |
| :JC =END | IF = 1, JUMP TO END |
| :T FW10 | IF = 0, TRANSFER MEASURED VAL. TO FW 10 |
| :T PB128 | INITIATE SAMPLING |
| END: | (1ST VAL. IS INVALID FOLLOWING RESTART) |
| :BE | |

## 10.8 Principle of Operation of Analog Output Modules

The CPU processes the digital values that the analog output modules convert to the required voltages or currents. Various floating modules cover individual voltage and current ranges.

### Signal Interchange between CPU and Module

The CPU transfers a digital value to the module's memory under a specified address. The user starts the transfer via FB251 or "T PB" or "T PW" operation.
Block diagram10-26 illustrates the principle of operation of the 470 analog output module.



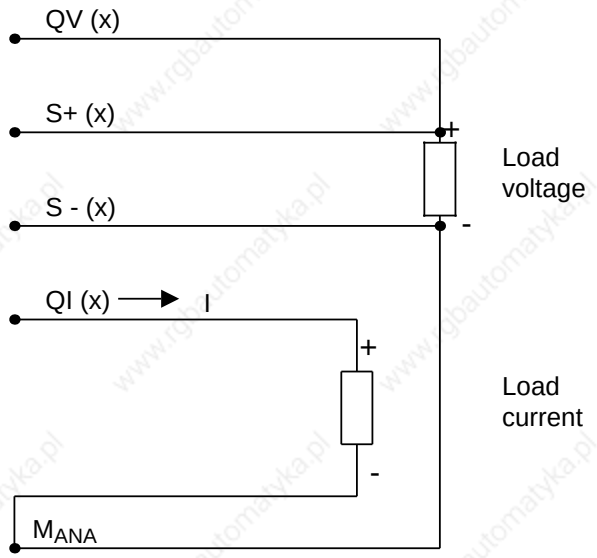

MUX Multiplexer
D/A Digital-analog converter

**Figure 10-26. Block Diagram with Signal Interchange between CPU and a 470 Analog Output Module**

### 10.8.1 Connecting Loads to Analog Output Modules

When loads are connected to analog output modules, the voltage is measured directly across the load via high-resistance sensing lines (S+/S -). The output voltage is then corrected so that the load voltage is not falsified by voltage drops on the lines.

In this way it is possible to compensate voltage drops of up to 3 V per line.
Figure 10-27 shows the design of this circuit.

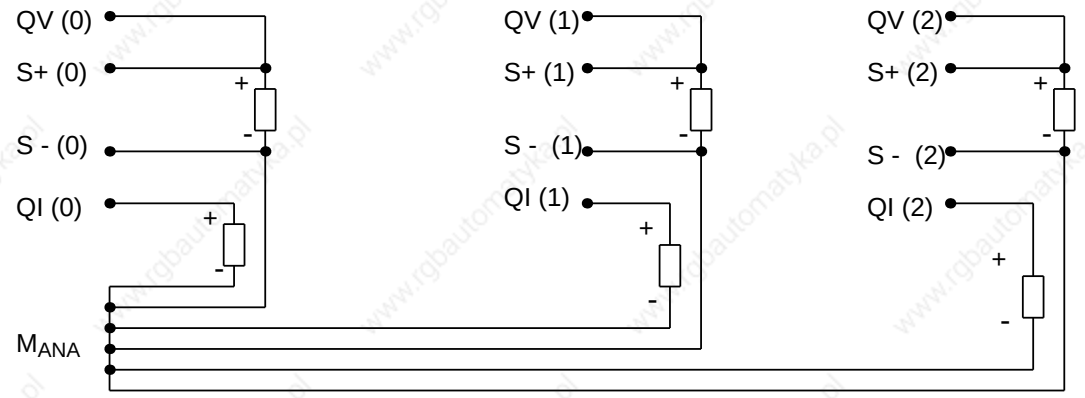



| | | | |
|---|---|---|---|
| QV | (x) | = | Analog output voltage (QV=Output voltage) |
| QI | (x) | = | Analog output current (QI=Output current) |
| S+ | (x) | = | Sensing line+ (S+=Sensing line+) |
| S - | (x) | = | Sensing line- (S -=Sense line -) |
| $M_{ANA}$ | | = | Ground terminal of the analog component |
| | x | = | Channel no. (0 to 7) |

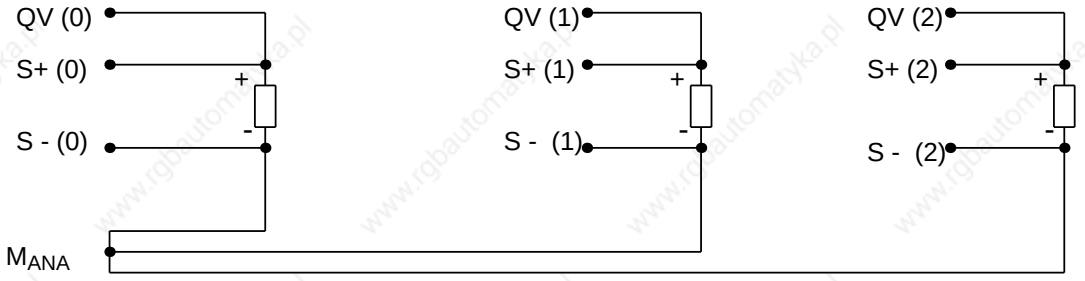
**Figure 10-27. Connecting Loads**

**Connecting Loads to Current and Voltage Outputs**

Figure 10-28 shows how to wire the analog output module.




6ES5 470-7LAxx
6ES5 470-7LCxx




6ES5 470-7LBxx

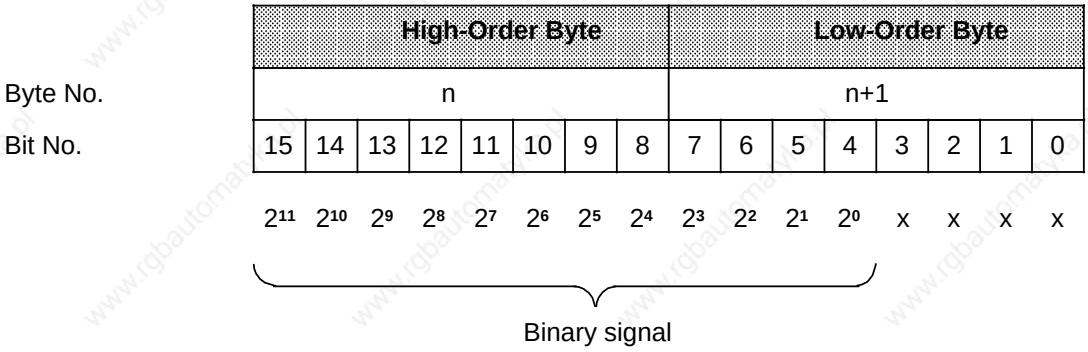
**Figure 10-28. Connecting Loads to Current and Voltage Outputs**

## Note

If voltage outputs are not used, or if only current outputs are connected, jumpers must be inserted in the front connector for the unused voltage outputs. To do this, connect QV (x) to S+(x) and S - (x) to $M_{ANA}$.
Unused current outputs remain open.

## 10.8.2 Digital Representation of an Analog Value

The CPU uses two bytes to represent the value of an output channel.
Figure 10-29 explains the individual bits:




x represents an irrelevant bit

**Figure 10-29. Representation of an Analog Output Signal in Digital Form**

> **Note**
>
> For the two's complement, bit $2^{11}$ indicates the sign
> (0 equals a positive value, 1 a negative value).

Table 10-36 lists the output voltages or currents of the individual 470-... analog output modules.

**Table 10-36. Analog Output Signals**

| Units | Output Voltages and Currents of the Modules | | | | Digitized Output Value* | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | -7LA/B12 in V | -7LA12 in mA | -7LC12 in V | -7LC12 in mA | $2^{11}$ | $2^{10}$ | $2^9$ | $2^8$ | $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ | |
| +1280 | +12.5 | 25.0 | 6.0 | 24.0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Over-range |
| +1025 | +10.0098 | 20.0195 | 5.004 | 20.016 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | |
| **+1024** | **+10.0** | **20.0** | **5.0** | **20.0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **+1023** | **+9.99** | **19.98** | **4.995** | **19.98** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | |
| **+512** | **+5.0** | **10.0** | **3.0** | **12.0** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **+256** | **+2.5** | **5.0** | **2.0** | **8.0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **+128** | **+1.25** | **2.5** | **1.5** | **6.0** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **+64** | **+0.625** | **1.25** | **1.25** | **5.0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **+1** | **+0.0098** | **0.0195** | **1.004** | **4.016** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **Nominal value** |
| **0** | **+0.0** | **0.0** | **1.0** | **4.0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **-1** | **-0.0098** | **0.0** | **0.996** | **3.984** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | |
| **-64** | **-0.625** | **0.0** | **0.75** | **3.0** | **1** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **-128** | **-1.25** | **0.0** | **0.5** | **2.0** | **1** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **-256** | **-2.5** | **0.0** | **0.0** | **0.0** | **1** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **-512** | **-5.0** | **0.0** | **-1.0** | **0.0** | **1** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| **-1024** | **-10.0** | **0.0** | **-3.0** | **0.0** | **1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | |
| -1025 | -10.0098 | 0.0 | -3.004 | 0.0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | Over-range |
| -1280 | -12.5 | 0.0 | -5.0 | 0.0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

* The insignificant bits have been omitted

## 10.9 Analog Value Matching Blocks FB250 and FB251

These blocks match the nominal range of an analog module to a normalized range that you can specify.

### Reading and Scaling an Analog Value - FB250 -

Function block FB250 reads an analog value from an analog input module and outputs a value XA in the scaled range specified. Define the desired range using the "upper limit" (OGR) and "lower limit" (UGR) parameters.
Specify the type of analog value representation (channel type) in the KNKT parameter (see Section 10). The BU parameter is set when the analog value exceeds the nominal range.

### Call and Parameter Assignments

| Parameter | Description | Type | Data Type | Assignment | STL |
|---|---|---|---|---|---|
| BG | Module address | D | KF | 128 ... 224 | : JU FB 250<br>NAME : RLG:AE<br>BG :<br>KNKT :<br>OGR :<br>UGR :<br>EINZ :<br>XA :<br>FB :<br>BU :<br>TBIT : |
| KNKT | KN= Channel number<br>KT= Channel type | D | KY | KY =x,y<br>x =0 to 15<br>y =3 to 6<br>3: absolute value representation (4 to 20 mA)<br>4: unipolar representation<br>5: absolute value, bipolar<br>6: bipolar fixed-point number (two's complement) | |
| OGR | Upper limit of the output value | D | KF | - 32768 to +32767 | |
| UGR | Lower limit of the output value | D | KF | - 32768 to +32767 | |
| EINZ | Selective sampling | I | BI | Selective sampling is triggered with "1". | |
| XA | Output value | Q | W | Standardized analog value is "0" on wirebreak. | |
| FB | Error bit | Q | BI | "1" on wirebreak, illegal channel or slot number, or illegal channel type | |
| BU | Range violation | Q | BI | "1" when nominal range is exceeded | |
| TBIT | Activity bit of the function block | Q | BI | The function block is in the process of selective sampling when the signal state is "1". | |

Scaling:
Function block FB250 converts the value read linearly to accord with the upper and lower limiting values using the following formula:

For channel type 3 (absolute value 4 to 20 mA):

$$XA = \frac{UGR \cdot (2560\text{-}xe) + OGR \cdot (xe\text{-}512)}{2048}$$

For channel type 4 (unipolar representation):

$$XA = \frac{UGR \cdot (2048\text{-}xe) + OGR \cdot xe}{2048}$$

For channel type 5 and 6 (bipolar representation):

$$XA = \frac{UGR \cdot (2048\text{-}xe) + OGR \cdot (xe\text{+}2048)}{4096}$$

Where XA is the value output by the FB and
xe is the analog value read from the module.

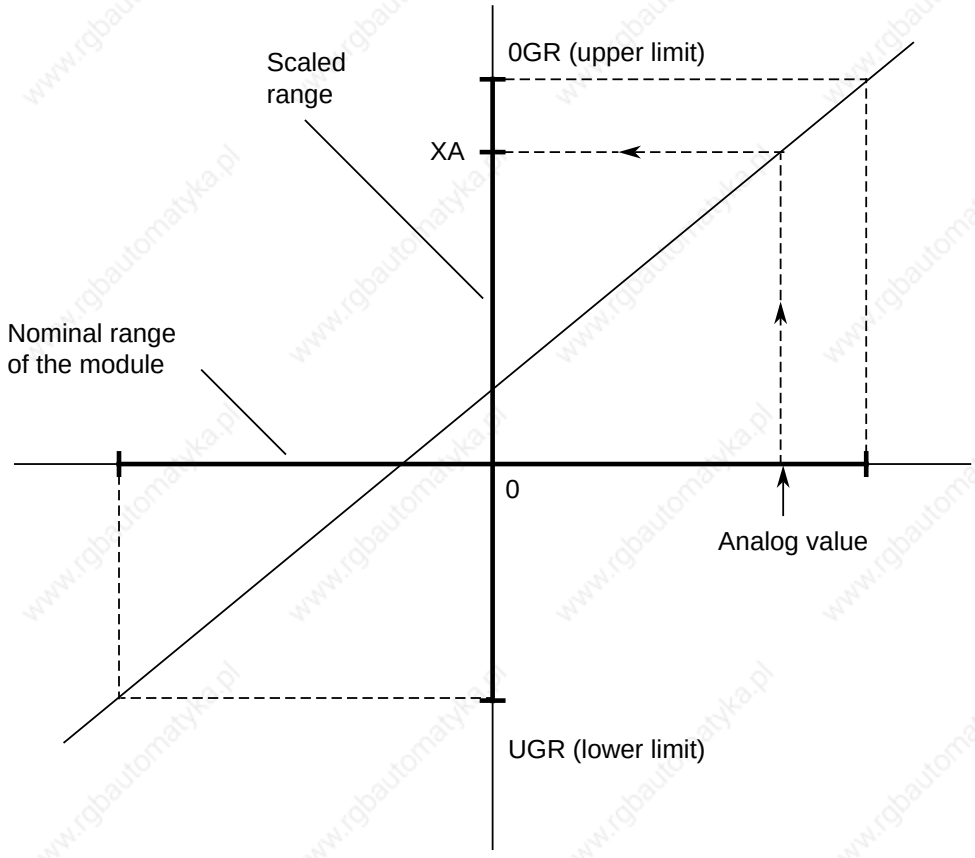



**Figure 10-30. Schematic Representation of Conversion**

**Selective Sampling**

FB 250 permits reading of an analog value with selective sampling. Setting the "EINZ" parameter to "1" causes the analog input module to convert the analog value of the selected channel to a digital value immediately. During conversion (approximately 60 msec.), no further sampling operations involving this module may be initiated. Consequently, the function block that is presently active sets the TBIT to "1" until the converted value is read in. The TBIT is reset upon completion of selective sampling is terminated.

## Note

The 466-3LA11 analog input module has no overrange! When the measured value reaches the nominal range limit, the overflow bit is set.
If you use standard FB250 (NAME:RLG:AE) of CPUs 941 to 944 for reading in the analog values in the case of the 466 module, you must take account of the following:

- Bipolar measuring range:
  If the measured value reaches the nominal range limit, the BU (range violation) parameter is set. The analog value read in is then ***invalid***!
- Unipolar measuring range:
  If the measured value reaches the midpoint of the nominal range, the BU parameter is set. The measured value read in is then ***valid***!

**Outputting an Analog Value -FB251-**

Use function block FB251 to output analog values to analog output modules. Specify the module's type of analog representation (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 relevant module using the following formula:

For channel type 0 (unipolar representation):

$$xa = \frac{1024 \cdot (XE\text{-}UGR)}{OGR\text{-}UGR}$$

For channel type 1 (bipolar representation):

$$xa = \frac{1024 \cdot (2 \cdot XE\text{-}OGR\text{-}UGR)}{OGR\text{-}UGR}$$

Where XE is the digital value specified in the function block and
xa is the value output to the module.

**Calling and Initializing -FB251-**

| 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<br>NAME : RLG:AA<br>XE :<br>BG :<br>KNKT :<br>OGR :<br>UGR :<br>FEH :<br>BU : |
| BG | Module address | D | KF | 128 to 240 | |
| KNKT | KN= channel number<br>KT= channel type | D | KY | KY =x,y<br>x =0 to 7<br>y =0;1<br>0: unipolar representation<br>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,<br>for illegal channel<br>or slot number,<br>or illegal channel type | |
| BU | Analog value to be output exceeds UGR or OGR | Q | BI | At "1, "XE is outside the range (UGR;OGR).<br>XE assumes the limit value | |

## 10.10 Example of Analog Value Processing

**Problem Definition:**
A closed container contains a liquid. It should be possible to read the current liquid level on an indicating instrument whenever required. A flag is to be set when the liquid level reaches a specified limiting value.

- A 0 - 20 mA transducer transmits the liquid level signal (between 0 and 10 m) to a 6ES5 460-7LA12 (460 AI) analog input module.
- The analog input module converts the analog current values into digital units (0 - 2048 units), which can be postprocessed by the S5-115U's application program.
- The application program compares the values with a limiting value (max. permissible liquid level), sets a flag if necessary, and sends these values to a 6ES5 470-7LB12 (AO 470) analog output module.
- The analog output module reconverts the values into voltages (0 - 10 V). In response to these voltages, the needle on the analog display swings proportionally to the liquid level.

Figure 10-31 shows the system configuration.

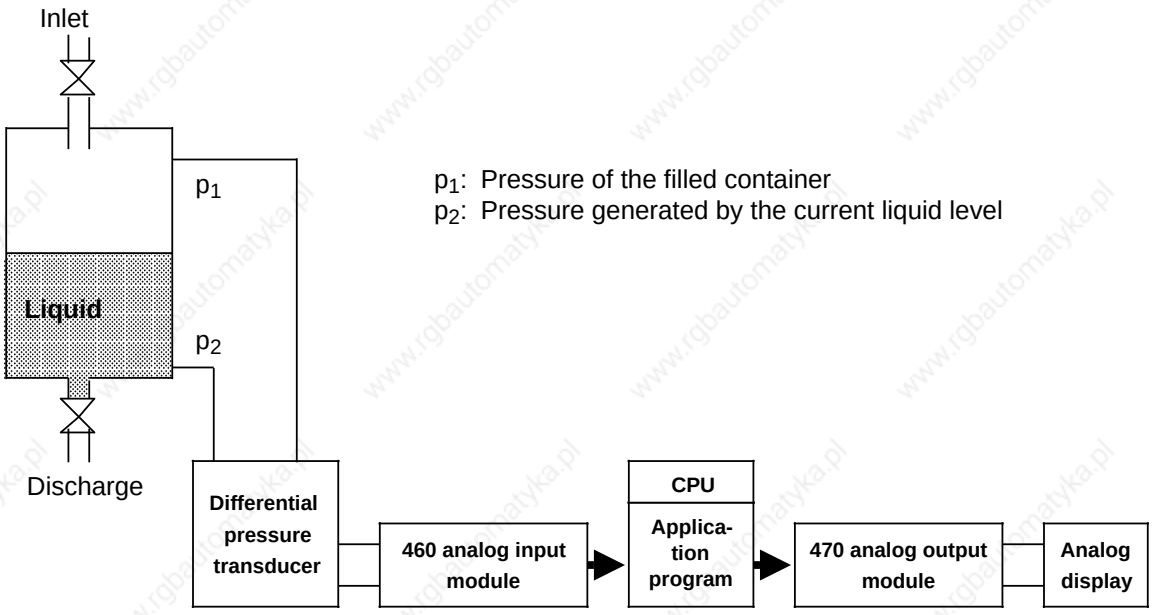



**Figure 10-31. Example of Analog Value Processing**

**Startup Procedures**

460 Analog Input Module:

Connect the transducer directly to the front connector on the AI 460 (Terminals: MO+, MO -). The transducer supplies values between 0 and 20 mA, 0 mA corresponding to a liquid level of 0.00 meters and 20 mA to the maximum liquid level, which is 10.00 meters.
Plug a±20 mA range card (6ES5 498-1AA41) into the AI 460.
A digital value between 0 and 2048 units, which is subsequently processed by the application program, is then present at the output of the analog input modules's internal ADC (see Figure 10-32).

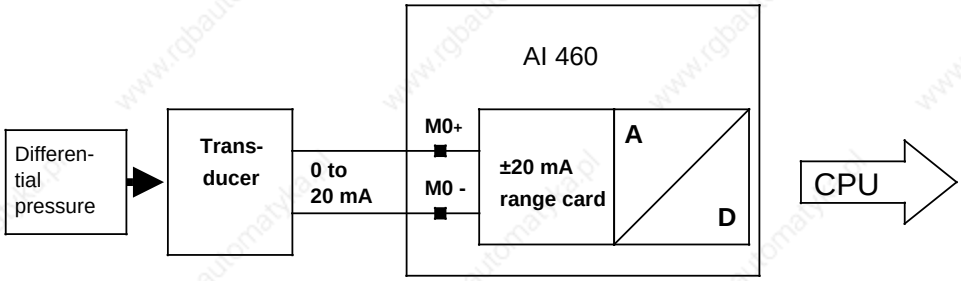



**Figure 10-32. Function of the 460 Analog Input Module**

Set the mode selectors at the rear of the module as follows (Figure 10-33):

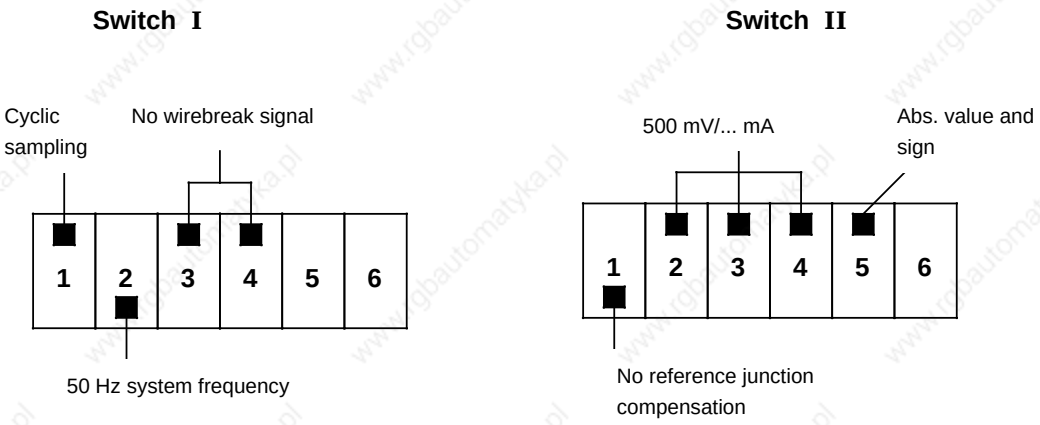



**Figure 10-33. Setting Mode Selectors I and II**

470 Analog Output Module:

Connect the indicating instrument directly via the module's front connector (pins: QV0, S + 0, S - 0, $M_{ANA}$).
The analog output modules outputs a voltage between 0 and 10 V to the indicating instrument, thus making it possible to read the liquid level as an analog value (Figure 10-34).

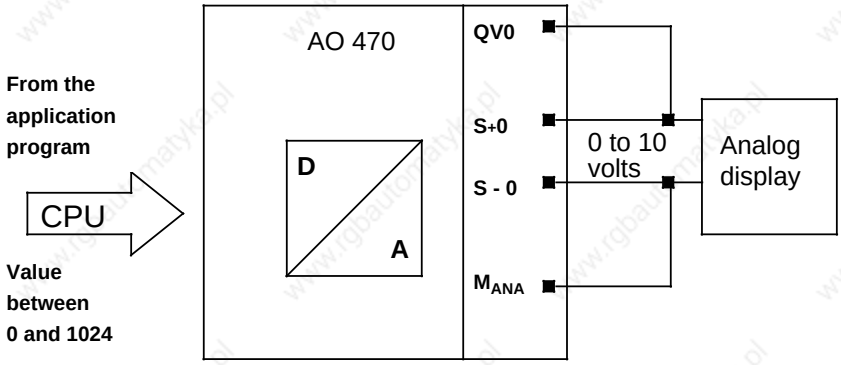



**Figure 10-34. Function of the 470 Analog Output Module**

**Program Structure**

Call and initialize "Read analog value" function block FB250 (for conversion to a range of from 0 to 1000 cm [XA parameter]).
Generate the limiting value (PB9).
A flag (F 12.6) is set when the liquid level exceeds 900 cm.
Call and initialize "Output analog value" function block FB251 (for conversion of a value in the range from 0 to 1000 cm [XE parameter] into a value between 0 and 1024 units for the AO 470).

Integral function blocks FB250 and FB251 are discussed in detail in Section 10-9 under the heading "Integral Function Blocks".

| PB1 STL | Description |
|---|---|
| :JU FB 250 | |
| NAME :RLG:AE | |
| BG :KF +128 | MODULE STARTING ADDR: 128 (WHEN SLOT ADDRESSING IS FIXED: SLOT 0) |
| KNKT :KY 0,4 | CHANNEL NO.: 0; UNIPOLAR REPRESENTATION: 4 |
| OGR :KF +1000 | PHYSICAL MEASURING RANGE: |
| UGR :KF +0 | 0<XA<1000CM |
| EINZ :F 12.0 | RELEVANT ONLY FOR SELECTIVE SAMPLING (SET IN EXAMPLE FOR: CYCLIC SAMPLING) |
| XA :FW 10 | IN FW 10: XA VALUE 0<XA<1000CM |
| FB :F 12.1 | RELEVANT ONLY WHEN SET FOR WIREBREAK SIGNAL |
| BU :F 12.2 | IF LEVEL > 1000CM, BU = 1. |
| TBIT :F 12.3 | RELEVANT ONLY FOR SELECTIVE SAMPLING. |

| PB1 STL (cont.) | Description |
|---|---|
| :JU PB 9<br>:JU FB 251<br>NAME :RLG:AA<br>XE :FW 10<br>BG :KF +160<br><br>KNKT :KY 0,0<br>OGR :KF +1000<br>UGR :KF +0<br>FEH :F 12.4<br>BU :F 12.5<br>:BE | GENERATE LIMITING VALUE<br>OUTPUT ANALOG VALUE<br><br>XA (FB 250) = XE (FB 251)<br>MODULE STARTING ADDR.; 160 (FIXED SLOT ADDRESSING: SLOT 1)<br>CHANNEL NO.: 0; UNIPOLAR REPRESENTATION: 0<br>PHYSICAL MEASURING RANGE:<br>0<XA<1000CM<br>WHEN UGR = OGR, FEH = 1<br>WHEN XA<UGR OR XA>OGR, BU = 1. |

| PB9 STL | Description |
|---|---|
| :L KF +900<br>:L FW 10<br>:<=F<br>:=F 12.6<br><br>:BE | MAX. VAL. FOR LIQUID LEVEL<br>MEASURED VALUE<br>MEASURED VALUE > 900 ?<br>IF YES, F 12.6 = INITIATE REACTION<br>IN SAME PROGRAM CYCLE. |

## 11 Integral Blocks


11.1 Integral Function Blocks . . . . . 11- 2
11.1.1 Conversion Blocks . . . . . 11- 2
11.1.2 Arithmetic Blocks . . . . . 11- 3
11.1.3 Data Handling Blocks . . . . . 11- 5
11.1.4 The Integral "COMPR" Block . . . . . 11- 28
11.1.5 Integral FB "DELETE" . . . . . 11- 30

11.2 Organization Blocks . . . . . 11- 32
11.2.1 OB31 Scan Time Triggering . . . . . 11- 32
11.2.2 OB160 Variable Time Loop . . . . . 11- 32
11.2.3 OB251 PID Control Algorithm . . . . . 11- 33
11.2.4 OB254 Read In Digital Input Modules (CPU 944 Only) . . . . . 11- 45
11.2.5 OB255 Transfer the Process Output Image (PIQ) to the Output Modules (CPU 944 Only) . . . . . 11- 45

11.3 DB1: Initializing Internal Functions . . . . . 11- 46
11.3.1 Configuration and Default Settings for DB1 . . . . . 11- 46
11.3.2 Setting the Addresses for the Parameter Error Code in DB1 (An example of how to set the parameters correctly) . . . . . 11- 47
11.3.3 How to Assign Parameters in DB1 . . . . . 11- 48
11.3.4 Rules for Setting Parameters in DB1 . . . . . 11- 49
11.3.5 How to Recognize and Correct Parameter Errors . . . . . 11- 50
11.3.6 Transferring the DB1 Parameters to the PLC . . . . . 11- 53
11.3.7 Reference Table for Initializing DB1 . . . . . 11- 54
11.3.8 DB1 Programming Example . . . . . 11- 56

## Figures


11-1. Format of the Job Status Word . . . . . 11- 16
11-2. Format of the "PAFE" Byte . . . . . 11- 19
11-3. Block Diagram of the PID Controller . . . . . 11- 33
11-4. Estimating the Dominant System Time Constant ($T_{RKdom}$) . . . . . 11- 39
11-5. Process Schematic . . . . . 11- 40
11-6. Initialization Error Code and its Significance . . . . . 11- 52
11-7. DB1 with Parameter Error . . . . . 11- 53


## Tables


11-1. Overview of Integral Blocks . . . . . 11- 1
11-2. List of Data Handling Block Parameters . . . . . 11- 5
11-3. QTYP/ZTYP Parameters . . . . . 11- 9
11-4. Ready Delay Times of the CPs and IPs . . . . . 11- 10
11-5. Basic Format of the Doubleword for the Job Status . . . . . 11- 15
11-6. Description of the Error Bits . . . . . 11- 16
11-7. Accessing the Job Status Word . . . . . 11- 17
11-8. Accessing the Length Word . . . . . 11- 19
11-9. Error Bits Set by FB239 (ERR Parameter) . . . . . 11- 31
11-10. Description of the Control Bits in Control Word STEU . . . . . 11- 35
11-11. Format of the Transfer Block . . . . . 11- 37

# 11 Integral Blocks

The following are integrated in the operating system of the central processing units:

- Some standard function blocks
- Some organization blocks
- A default DB1 for initializing internal functions.

Integral function blocks and organization blocks are programmed in machine language and so execute at high speed. They do not use space in internal program memory.

Integral blocks are called in the control program like all other blocks; they can only be interrupted by process interrupts.

(Integral) function blocks which can be called in the control program for special functions are the subject of this chapter. Those blocks which the operating system of the CPU calls automatically when specific events occur (e.g. programming errors, PLC faults) are not dealt with here (see Chapter 7).

**Table 11-1. Overview of Integral Blocks**

| Block | | | Call Length (in Words) | Execution time (in Millise-conds) | Function |
|---|---|---|---|---|---|
| Type | No. | Name | | | |
| FB | 238 | COMPR [1] | 4 | < 0.6 | Compress PLC memory |
| FB | 239 | DELETE | 5 | < 0.8 | Delete block |
| FB | 240 | COD : B4 | 5 | < 0.8 | 4-tetrad BCD code converter |
| FB | 241 | COD : 16 | 6 | < 1.0 | 16-bit fixed-point converter |
| FB | 242 | MUL : 16 | 7 | < 1.0 | 16-bit binary multiplier |
| FB | 243 | DIV : 16 | 10 | < 2.3 | 16-bit binary divider |
| FB | 244 | SEND [2] | | | Send data |
| FB | 245 | RECEIVE [2] | | | Receive data |
| FB | 246 | FETCH | | < 4 | Fetch data |
| FB | 247 | CONTROL | | approx. 0.6 | Monitor job processing |
| FB | 248 | RESET | | < 4 | Delete job |
| FB | 249 | SYNCHRON | | 6.5 ms to 10 s | Initialize interface |
| FB | 250 | RLG : AE | 11 | 2.0 | Read analog value (see Chapter 10) |
| FB | 251 | RLG : AA | 9 | 5.2 | Output analog value (see Chapter 10) |
| OB | 31 | | | < 0.14 | Restart scan time |
| OB | 160 | | | | Variable time loop |
| OB | 251 | | | 2.0 | PID control algorithm |
| OB | 254 [3] | | | | Read digital inputs |
| OB | 255 [3] | | | | Tranfer PIQ to outputs |
| DB | 1 | | | | Parameterize internal functions |

**1** Execution time not including block-dependent compression

**2** The execution time depends on the size of the data block to be transferred (see Section 11.1.3, "Frame Size")

**3** See Section 2.6.2

## 11.1 Integral Function Blocks

Integral function blocks can be devided into various groups according to function.

### 11.1.1 Conversion Blocks

Use blocks FB240 and FB241 to convert numbers in BCD code to fixed-point binary numbers and vice versa.

**Code Converter: B4 -FB240-**

Use function block FB240 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".

Call and Parameter Assignments

| Parameter | Type | Data Type | Assignment | Meaning | STL |
|---|---|---|---|---|---|
| BCD | I | W | - 9999 to+9999 | BCD number | : JU FB 240<br>Name : COD : B4<br>BCD :<br>SBCD :<br>BINARY: |
| SBCD | I | Bi | "1" for "-"<br>"0" for "+" | Sign of the BCD number | |
| BINARY | Q | W | 16 bits "0" or "1" | Binary number | |

**Code Converter: 16 -FB241-**

Use function block FB241 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 transferred to a 16-bit word before conversion.

Call and Parameter Assignments

| Parameter | Type | Data Type | Assignment | Meaning | STL |
|---|---|---|---|---|---|
| BINARY | I | W | - 32768 to+32767 | Binary number | : JU FB 241<br>Name : COD : 16<br>BINARY:<br>SBCD :<br>BCD2 :<br>BCD1 : |
| SBCD | Q | Bi | "1" for "-"<br>"0" for "+" | Sign of the BCD number | |
| BCD2 | Q | By | 2 tetrads | BCD number tetrads 4 and 5 | |
| BCD1 | Q | W | 4 tetrads | BCD number tetrads 0 to 3 | |

## 11.1.2 Arithmetic Blocks

Use function blocks FB242 and FB243 to multiply and divide.

**Multiplier : 16 -FB242-**

Use function block FB242 to multiply one fixed-point binary number (16 bits) by another. The product is represented by two fixed-point binary numbers (16 bits each). The result is also scanned for zero. An eight-bit number must be transferred to a 16-bit word prior to multiplication.

Call and Parameter Assignments

| Parameter | Type | Data Type | Assignment | Meaning | STL |
|---|---|---|---|---|---|
| Z1 | I | W | - 32768 to+32767 | Multiplier | : JU FB 242<br>Name : MUL : 16<br>Z1 :<br>Z2 :<br>Z3=0 :<br>Z32 :<br>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-order word | |
| Z31 | Q | W | 16 bits | Product, low-order word | |

**Divider: 16 -FB243-**

Use function block FB243 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 transferred to a 16-bit word prior to division.

Call and Parameter Assignment

| Parameter | Type | Data Type | Contents | Meaning | STL |
|---|---|---|---|---|---|
| Z1 | I | W | - 32768 to+32767 | Dividend | : JU FB 243<br>Name : DIV : 16<br>Z1 :<br>Z2 :<br>OV :<br>FEH :<br>Z3=0 :<br>Z4=0 :<br>Z3 :<br>Z4 : |
| Z2 | I | W | - 32768 to+32767 | Divisor | |
| OV | Q | Bi | "1", if overflow | Overflow indicator | |
| FEH | Q | Bi | "1" for division by zero | | |
| Z3=0 | Q | Bi | "1": quotient is zero | Scan for zero | |
| Z4=0 | Q | Bi | "1": remainder is zero | Scan for zero | |
| Z3 | Q | W | 16 bits | Quotient | |
| Z4 | Q | W | 16 bits | Remainder | |

### 11.1.3 Data Handling Blocks

Function blocks FB244 to FB249 make it possible to use communications processors and intelligent I/O modules. These “data handling blocks“ control data exchange between such modules and the CPU.

Data handling blocks offer the following advantages:

- They take up no space in the user memory.
- Transfer from diskette is not necessary.
- They have a short runtime.
- No flag, timer or counter areas are needed.

#### Parameters

Data handling blocks use the parameters listed in Table 11-2.

**Table 11-2. List of Data Handling Block Parameters**

| Name | | Meaning |
|---|---|---|
| SSNR | : | Interface number (page number) |
| A-NR | : | Job number |
| ANZW | : | Job status word (double word) |
| QTYP/ZTYP [1] | : | Type of data source or data destination |
| DBNR [1] | : | Data block number |
| QANF/ZANF [1] | : | Relative start address within a type |
| QLAE/ZLAE [1] | : | Length of source or destination data |
| PAFE [2] | : | Parameter assignment error |
| BLGR | : | Frame size |

1 If these parameters are not needed for a call (e.g., for the ALL function), you can skip them when initializing the block by pressing the <CR> key.
2 PAFE must be directly initialized.

**Parameter Description**

The formal operands that you must supply when using data handling blocks are explained below.

**SSNR - Interface Number**

The SSNR parameter specifies the logical number of the interface (page) to which a particular job refers.

| Parameter Type | Parameter Format | Assignment | | |
|---|---|---|---|---|
| Data (byte) | KY | KY= | x,y | |
| | | | x=0 | Direct initializing |
| | | | y=0 to 255 | Interface number (page address) |
| | | | x 0 | Indirect initialization |
| | | | y=0 to 255 | Data word number. The SSNR, A-NR, and ANZW parameters are stored beginning with the next data word in the current DB. |

**A-NR - Job Number**

The jobs for an interface are characterized by this number.

| Parameter Type | Parameter Format | Assignment | | |
|---|---|---|---|---|
| Data (byte) | KY | KY= | x,y | Parameter x is ignored<br><br>"y" represents the job number. |
| | | | y=0 | ALL function [1] |
| | | | y=1 to 223 | Direct function<br>Number of the job to be executed. [2] |

1 The "ALL" function is not permitted for the FETCH block.
2 Refer to the SINEC L1 Local Area Network manual for an explanation of the individual job numbers.

**ANZW - Job Status Word**

Use this parameter to specify the address of a double word (DW * n/DW n+1 or FW n and FW n+2) that indicates the processing status of a particular job.

| Parameter Type | Parameter Format | Assignment | |
|---|---|---|---|
| Address (word) | W | x=0 to 255 | Address of the job status word for direct initialization<br>Permissible areas: DW, FW |

**QTYP/ZTYP - Type of Data Source or Data Destination**

Assign these parameters ASCII characters that specify the type of data source (for SEND) or data destination (for RECEIVE or FETCH).

| Parameter Type | Parameter Format | Assignment | |
|---|---|---|---|
| Data (character) | KS | KS = | DB, QB, IB, FY, TB, CB, AS, PB<br>Direct initialization: The specification of the data source (or data destination) is specified directly in the QTYP/ZTYP, DBNR, QANF/ZANF, QLAE, ZLAE parameters. |
| | | KS = NN | No parameter assignment: Data source (or data destination) specifications are located in the job request block in the job. |
| | | KS = RW, XX | Indirect initialization: Specifications for data source (or data destination) are located in a data area specified by the |

* DW refers to the current data block

**DBNR - Data Block Number**

If DB, RW, or XX were assigned to the parameters QTYP/ZTYP, the DBNR parameter must specify the number of the required data block.

| Parameter Type | Parameter Format | Assignment |
|---|---|---|
| Data (byte) | KY | KY = 0, y<br>y =2 to 255 Number of the data block containing the data |

**QANF/ZANF - Start Address of the Source or Destination Data Block**

When initialization is indirect (QTYP/ZTYP=RW or XX), specify the number of the DW at which the parameter block begins.
When initialization is direct, QANF/ZANF refers to the specified area.

| Parameter Type | Parameter Format | Assignment |
|---|---|---|
| Data (fixed-point No.) | KF | Permissible range (see Table 11-3) |

**QLAE/ZLAE - Length of the Source or Destination Data Frame**

When initialization is direct, the source or destination type specification is understood to be the number of either bytes or words.

| Parameter Type | Parameter Format | Assignment |
|---|---|---|
| Data (constant) | KH<br>KF | Permissible range (see Table 11-3)<br>- 1 : The "joker length" - 1 means the following:<br>• for RECEIVE: As much data as the transmitter sends or as much as area limitations permit.<br>• for SEND: Data is transmitted until a particular area boundary is reached. |

**Summary:**

**Table 11-3. QTYP/ZTYP Parameters**

| | QTYP/ZTYP Description | DBNR Meaning Area Permitted | QANF/ZANF Meaning Area Permitted | QLAE/ZLAE Meaning Area Permitted |
|---|---|---|---|---|
| **NN** | No source/destination parameters in the block. Parameters have to be in the CP. | Irrelevant | Irrelevant | Irrelevant |
| **XX** | Indirect addressing: parameters are stored in the data block (specified with DBNR and QANF). | DB in which the source/destination parameters are stored 2 to 255 | DW number with which the parameters begin 0 to 2047 | Irrelevant |
| **RW** | Indirect addressing without data exchange. Source or destination parameters are stored in a DB.[1] | DB in which the source/destination parameters are stored 2 to 255 | DW number with which the parameters begin 0 to 2047 | Irrelevant |
| **DB** | Source/destination data from/to the data block in main memory | DB from which source data are taken or to which destination data are transferred 2 to 255 | DW number beginning with which the data is to be read or written 0 to 2047 | Length of the source/ destination data in words 1 to 2048 |
| **MB** | Source/destination data from/to the flag area | Irrelevant | Flag byte number beginning with which the data is to be read or written 0 to 255 | Length of the source/ destination data in bytes 1 to 255 |
| **AB** | Source/destination data from/to the process output image (PIQ) | Irrelevant | Output byte number beginning with which the data is to be read or written 0 to 127 | Length of the source/ destination data in bytes 1 to 128 |
| **EB** | Source/destination data from/to the process input image (PII) | Irrelevant | Input byte number beginning with which the data is to be read or written 0 to 127 | Length of the source/ destination data in bytes 1 to 128 |
| **PB** | Source data from input modules<br><br>Destination data to output modules | Irrelevant | Peripheral byte number beginning with which the data is to be read or written<br>0 to 127 digital I/Os<br>128 to 255 analog I/Os | Length of the source/ destination data in bytes 1 to 256 |
| **ZB** | Source/destination data from/to counter locations | Irrelevant | Number of the counter location beginning with which the data is to be read or written 0 to 127 | Length of the source or destination data in words (counter location=1 word) 1 to 128 |
| **TB** | Source/destination data from/to timer locations | Irrelevant | Number of the timer location beginning with which the data is to be read or written 0 to 127 | Length of the source or destination data in words (timer location=1 word) 1 to 128 |
| **AS** | Source/destination data from/to memory locations absolute-addressed | Irrelevant | Absolute start address beginning with which the data is to be read or written 0 to +32767 - 32768 | Length of the source/destination data block in words 1 to 32767 $0000_H$ to $FFFF_H$ |

1 Assigning RW to ZTYP is not permitted for the RECEIVE block

**BLGR - Frame Size**

The BLGR parameter specifies the maximum size of the data frame that can be exchanged between a PLC and a CP during one pass of the data handling block (applies only to SYNCHRON).

| Parameter Type | Parameter Format | Assignment | | | |
|---|---|---|---|---|---|
| Data (byte) | KY | KY = 0,y | Frame Size | Execution Time CPU 941 - 943 (SEND and RECEIVE) | Execution Time CPU 944 (SEND and RECEIVE) |
| | | y =0 | 64 bytes ** | approx. 6.0 msec. | The execution time is as follows:<br>- in the case of processing without data inter-change: approx. 0.9 ms<br><br>- in the case of processing with data interchange: Total execution time: =4500 µs+Number of bytes (1.7 µs+ready delay time of the CP/IP*) |
| | | y =1 | 16 bytes | approx. 3.0 msec. | |
| | | y =2 | 32 bytes | approx. 4.5 msec. | |
| | | y =3 | 64 bytes | approx. 6.0 msec. | |
| | | y =4 | 128 bytes | approx. 10.0 msec. | |
| | | y =5 | 256 bytes | approx. 17.5 msec. | |
| | | y =6 | 512 bytes | approx. 30 msec. | |
| | | y =7 to 254 | same as for y=0 | | |

* See Table 11-4

****** The block uses the default parameter. (On the S5-115U, the frame size is set at 64 bytes).

**Table 11-4. Ready Delay Times of the CPs and IPs**

| Communications Processor | Ready Delay Time in µs |
|---|---|
| CP 524 | 1 |
| CP 525 | 3 |
| CP 526 | 3 |
| CP 530 | 3 to 130 |
| CP 535 | 3 |
| CP 551 | 3 |
| CP 552 | 3 |
| IP 252 | 10 |
| IP 246 | 1.5 |
| IP 247 | 1.5 |
| CP 527 | 3 |
| CP 5430 | 1 |
| CP 143 | 3 |

**PAFE - Parameter Assignment Error Byte**

For PAFE, specify a byte that is set if the block detects a parameter assignment error. The following can be parameter assignment errors:

- No such interface
- The QTYP/ZTYP, QANF/ZANF, or QLAE/ZLAE parameters were assigned incorrectly.

| Parameter Type | Parameter Format | Assignment |
|---|---|---|
| Address (byte) | BY | QB 0 to 127<br>FB 0 to 255 |

**Direct and Indirect Initialization**

The high-order byte of the SSNR parameter is the selection criterion for direct or indirect initialization:

- High-order byte of SSNR=0 means direct initialization
  SSNR, A-NR, ANZW or BLGR are specified directly in the block.
- High-order byte of SSNR 0 means direct initialization
  SSNR, A-NR, ANZW or BLGR are stored in the current data block, beginning with the data word specified in the low-order byte of the SSNR parameter.

SSNR and A-NR have the same data format (KY) in both cases. Representation formats are different for the job status word. While the address of the job status word is specified directly when initialization is direct (e.g., FW 100), an additional specification concerning the area in which the job status word is located must be made when initialization is indirect. This area is specified in ASCII code in the data word preceding the job status word.

FW means that the job status word is located in the flag area, DB that it is located in a data block.

In the next data word of the parameter area in the DB is the ANZW address in KY data format and, if ANZW is located in a data block, the block number (in the first byte of the KY format).

Examples:

**Direct Initialization of SSNR, A-NR, and ANZW**

- Job Status Word in the Flag Area

| Parameter Assignments | | Explanation |
|---|---|---|
| JU | FB 245 | |
| NAME : | RECEIVE | |
| SSNR : | KY 0,3 | The interface number is 3. |
| A-NR : | KY 0,100 | The job number is 100. |
| ANZW : | FW 240 | Flag words FW 240 and FW 242 are used as job status word. |

- Job Status Word in a Data Block

| Parameter Assignments | | Explanation |
|---|---|---|
| C | DB 47 | DB47 is activated. |
| JU | FB 247 | |
| NAME : | CONTROL | |
| SSNR : | KY 0,3 | The interface number is 3. |
| A-NR : | KY 0,100 | The job number is 100. |
| ANZW : | DW 40 | Data words DW 40 and DW 41 in DB47 are used as job status word. |

**Indirect Initialization of SSNR, A-NR and ANZW**

- Job Status Word as Flags

| Parameter Assignments | | Explanation |
|---|---|---|
| C | DB 44 | Open DB44 |
| JU | FB 244 | |
| NAME : | SEND | |
| SSNR : | KY 255,1 | ID for indirect initialization |
| . | | The data area for initialization begins with DW 1. |
| A-NR : | KY 0,0 | Irrelevant |
| ANZW : | FW 0 | Irrelevant |
| DB | 44 | |
| DW 1 | KY 0,1 | The interface number is 1. |
| DW 2 | KY 0,31 | The job number is 31. |
| DW 3 | KS MW | The job status word is in the flag area. |
| DW 4 | KY 0,200 | The job status word is represented in flag words FW 200 and FW 202. |

- Job Status Word in a Data Block

| Parameter Assignments | | Explanation |
|---|---|---|
| C | DB 24 | Open DB24 |
| JU | FB 244 | |
| NAME : | SEND | |
| SSNR : | KY 255,1 | ID for indirect initialization |
| . | | The data area for parameter assignment begins at |
| . | | data word 1. |
| A-NR : | KY 0,0 | Irrelevant |
| ANZW : | FW 0 | Irrelevant |
| DB | 24 | |
| DW 1 | KY 0,1 | The interface number is 1. |
| DW 2 | KY 0,31 | The job number is 31. |
| DW 3 | KS DB | The job status word is in a data block. |
| DW 4 | KY 222,10 | Address of the ANZW (DW 10 and DW 11 in DB222) |
| DB | 222 | |
| DW 10 | | Job status word |
| DW 11 | | |

**Indirect Initialization of SSNR and BLGR (SYNCHRON)**

| Parameter Assignments | | Explanation |
|---|---|---|
| C | DB 49 | Open DB49 |
| JU | FB 249 | |
| NAME : | SYNCHRON | |
| SSNR : | KY 255,100 | ID for indirect initialization<br>The data area for initialization begins with DW 100. |
| BLGR : | KY 0,0 | Irrelevant |
| DB | 49 | |
| DW 100 | KY 0,10 | The interface number is 10. |
| DW 101 | KY 0,6 | The block size is set at 512 bytes. |

**Indirect Initialization of QTYP/ZTYP, DBNR, QANF/ZANF, and QLAE/ZLAE**

When RW or XX is assigned to QTYP or ZTYP, the information for the source (or destination) is taken from a data area. The QANF parameter specifies the start address of this data area.
When XX is used for indirect initialization, enter the following data in the data block specified by "DBNR":

| Address in the Data Block | Parameter Type | Assignment | Explanation |
|---|---|---|---|
| QANF + 0 | KS | DB, QB, IB, FY, TB, CB, AS, NN | Type of source or destination |
| + 1 | KY | 2 to 255 * | Number of the DB for source or destination type DB (high-order byte=0) |
| + 2 | KF | 0 to 2047 | Start address of the source or destination area QANF/ZANF |
| + 3 | KF | 1 to 2048 | Length of the source or destination area |

***** Only for "DB"

For indirect initialization with RW, the data in the block with the "DBNR" number must contain the following information:

| Address in the Data Block | Parameter Type | Assignment | Explanation |
|---|---|---|---|
| QANF + 0 | KS | DB, QB, IB, FY, TB, CB, AS, NN | Source type specification |
| + 1 | KY | 2 to 255 * | Number of the DB for source type "DB" (high-order byte=0) |
| + 2 | KF | 0 to 2047 | Start address of the source data block |
| + 3 | KF | 1 to 2048 | Source data block length |
| + 4 | KS | DB, QB, IB, FY, TB, CB, AS, NN | Destination type specification |
| + 5 | KY | 2 to 255 * | Number of the DB for destination type "DB" (high-order byte=0) |
| + 6 | KF | 0 to 2047 | Start address of the destination data block |
| + 7 | KF | 1 to 2048 | Destination data block length |

* Only for "DB"

**Format and Meaning of the Job Status Word**

The job status word is used to store information on the status of jobs. Specify the address of the job status word when assigning parameters. Starting at this address, information can be read out and processed further.
Assign parameters to the ANZW such that a separate job status word is addressed for each job defined.
The job status word is part of a doubleword that is addressed by the ANZW parameter (see Table 11-5).

**Table 11-5. Basic Format of the Doubleword for the Job Status**

| Word No. | Meaning |
|---|---|
| n | Job status word |
| n + 1 | Length word |

### Job Status Word

The job status word is divided into four parts. Figure 11-1 explains the individual bits.

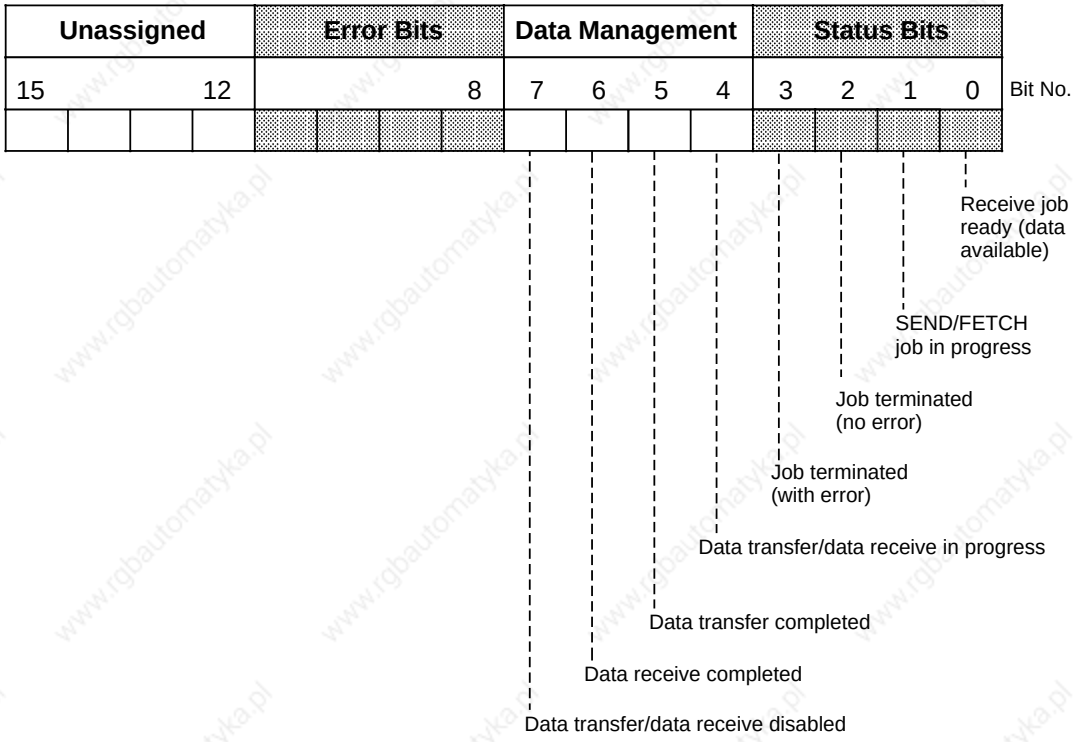



**Figure 11-1. Format of the Job Status Word**

### Description of the Error Bits

Error bits in the job status word are valid only if the "Job terminated with error" bit (bit 3) is set. Table 11-6 lists the possible errors.

**Table 11-6. Description of the Error Bits**

| Value of the Error Tetrads | Error |
|---|---|
| 0 | No error<br>If the "Job terminated with error" bit is set nonetheless, this means that the CP has set up the job again after a Cold Restart or a RESET. |
| 1 to 5 | PLC error, error codes the same as in PAFE |
| 6 to F | CP errors<br>CP-specific errors. Use the appropriate CP manual to determine the cause of the error. |

**Description of the Status and Data Management Bits**

The status bits and the data management bits can be set/reset and evaluated both by the user and via data handling blocks.
The table below shows the situations in which these bits are set or reset.

**Table 11-7. Accessing the Job Status Word**

| Bit No. | Set by | Reset/ Overwritten by | Evaluated by |
|---|---|---|---|
| 0 | DHB | DHB | • RECEIVE block (When this bit is set, handshaking with the CP is initiated).<br>• User (Scan to see if there is a frame). |
| 1 | DHB (as soon as the CP receives a job request) | DHB (when the CP serviced the job request) | • SEND/FETCH block (A new job is sent only after the old job has been processed).<br>• User (Scan to see if a new job is to be initiated). |
| 2 | DHB (if the job was completed without error) | DHB (if the job is reinitiated) | User (Scan to see if the job was completed without error). |
| 3 | DHB (if the job terminated with error). (The cause of the error is stored in the high-order byte of the job status word). | DHB (if the job is reinitiated) | User (Scan to see if the job was completed without error). |

**Table 11-7. Accessing the Job Status Word (Continued)**

| Bit No. | Set by | Reset/ Overwritten by | Evaluated by |
|---|---|---|---|
| 4 | DHB/SEND, RECEIVE (if data exchange has begun for a job Example: initiation with DIRECT function but exchange via ALL function) | DHB/SEND,RECEIVE (if data exchange is completed for a job) | User (Scan to see if the data frame has just been transferred). [1] |
| 5 | SEND block (if data has been transferred for a job) | • SEND block (if data transfer has been started for a new job)<br>• User (if an evaluation was made) | User (Scan to see if the dataset for a job has already been transferred to the CP and when a new dataset can be made available for a current job). |
| 6 | RECEIVE block (if data reception has been concluded for a job) | • RECEIVE block (if data transfer has been started for a new job)<br>• User (if an evaluation was made) | User (Scan to see if the data frame of a new job has already been transferred to the PLC and when a new data frame was transferred to the PLC for a job currently in progress). |
| 7 | User (Access of the SEND and RECEIVE blocks to an area is prevented at the first data field. Jobs already started are terminated). | User (The pertinent data area is enabled). | SEND-RECEIVE block (If the bit is set, the blocks do not execute any data traffic. Instead, they report an error to the CP). |

1 During data transfer between the CP and PLC, you can no longer modify the data for a job. This fact is not critical for small data packets since, in this case, data exchange can be handled in one block pass. However, large amounts of data can be transferred in blocks only. Consequently, data exchange can stretch over several program scans, depending on the frame size specified in the SYNCHRON block.

**Length Word:**

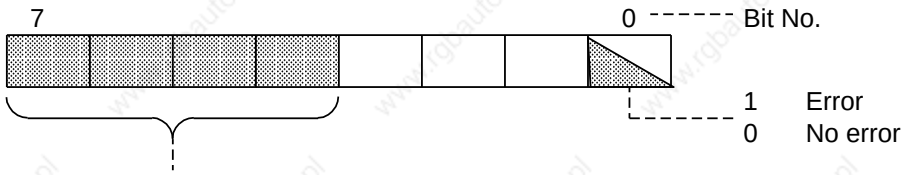
In the length word, the SEND and RECEIVE data handling blocks enter the amount of data (in bytes) already transferred for a particular job. For the ALL functions, the SEND and RECEIVE blocks enter the job number for which they were active in the current pass in the low-order byte. Job number "0" (empty run) means that no job was processed. Table 11-8 shows how the length word is acted upon.

**Table 11-8. Accessing the Length Word**

| Set by | Reset/ Overwritten by | Evaluated by |
|---|---|---|
| HTB/SEND, RECEIVE (during data exchange) The contents are calculated from the current number of transfers plus the quantity of (blocked) data already exchanged. | HTB/SEND, RECEIVE FETCH by over-writing during the next job. | User (If bit 2, 5, or 6 is set in the job status word, the current source or destination length is in the length word. If bit 3 is set, the length word contains the amount of data transferred prior to detection of an error). |

**"Parameter Assignment Error" (PAFE) Byte**

Only a flag byte is suitable as a condition code byte.
Various parameter assignment errors are reported in the PAFE byte (in the high-order tetrad). When assigning parameters, specify the address under which this information can be accessed. Figure 11-2 describes the individual bits.




0 No error
1 Incorrect ORG format, QTYP/ZTYP illegal (PLC or CP)
2 Area not available (no DB or illegal DB)
3 Area too small
4 Time-out (QVZ)/no access possible
5 Wrong job status word
6 No source or destination parameters for SEND/RECEIVE ALL (system error)
7 No such interface
8 Interface not ready
9 Interface overloaded
A Unassigned
B Illegal A-NR
C No acknowledgement or negative acknowledgement from the interface (CP)
D Parameter/BLGR illegal (first byte)
E Unassigned
F Unassigned

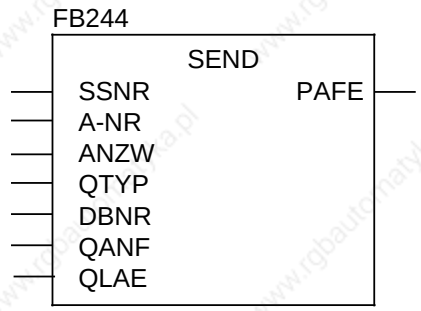
**Figure 11-2. Format of the "PAFE" Byte**

**The SEND Block - FB244 -**

FB244 requests that data be sent to a module with page addressing.
A distinction is made between two function modes:

- SEND All
  The function block is a substitute for direct memory access.
- SEND Direct
  Data is sent for a specific job.

**Calling Function Block FB244**

| STL | | | | CSF/LAD |
|---|---|---|---|---|
| | : JC | FB | 244 |  |
| NAME | : SEND | | | |
| SSNR | : | KY | 0,10 | |
| A-NR | : | KY | 0,32 | |
| ANZW | : | FW | 14 | |
| QTYP | : | KS | DB | |
| DBNR | : | KY | 0,10 | |
| QANF | : | KF | +1 | |
| QLAE | : | KF | +33 | |
| PAFE | : | FY | 13 | |
| | : *** | | | |



**Description of the SEND ALL Function**

For the SEND ALL function, the block requires the following parameters:

- SSNR - interface number
- A-NR - job number (assign "0")
- ANZW - job status word
- PAFE - parameter assignment error byte

All other parameters are irrelevant for this job. The CP uses the communications area to provide the following information:

- address of the job status word
- type of data
- amount of data
- start address of the data area

The following bits are evaluated or set/reset in the job status word for the pertinent job:

- data transfer disabled
- data transfer completed
- data transfer in progress

The SEND block enters the number of bytes transferred in the data word that follows the job status word.

The SEND block must be called in the control program in "ALL" mode at least once per interface when

- the CP can request data from a PLC on its own initiative, e.g., the CP 525 for display output or the CP 535 with the job mode "READ PASSIVE".
- a CP job is initiated with SEND DIRECT, but the CP asks the PLC for the data for this job via "background communications".
- the amount of data to be transmitted to the CP with SEND DIRECT is greater than the specified 1 frame size.

**Description of the SEND DIRECT Function**

The SEND DIRECT function works with the following parameters:

- SSNR - interface number
- A-NR - job number (assign " 0")
- ANZW - job status word
- PAFE - parameter assignment error byte
- QTYP - source type
- DBNR - data block number
- QANF - source start address
- QLAE - amount of source data

Normally, the SEND DIRECT function is called in the cyclic part of the control program. The block can be invoked in an interrupt service routine, but the job status word would not be updated cyclically in this case. This task must then be performed by the CONTROL block.

The following two conditions must be met to transfer data or to activate a SEND job:

- RLO "1" was forwarded to the function block
- The CP enabled the job. (The "SEND/FETCH in progress" bit of the condition code word is "0").

If RLO "0" is forwarded (empty run), only the job status word is updated.
If "NN" is entered in the QTYP parameter, the source parameters have to be stored in the CP. If not, the job is aborted with error.

Data interchange can proceed as follows:

- The requested data is transferred directly to the CP.
- The CP asks only for the job parameters.
- The amount of data to be transmitted is too large. The block transfers the parameters and the first data block to the CP. Then the CP requests the remaining data or an additional data frame from the PLC via the SEND ALL function.

For the block user, the operator interface is the same in all initiation words. However, in the last two cases, the instant of data transfer is postponed by at least one program cycle.

**Description of the WRITE Function**

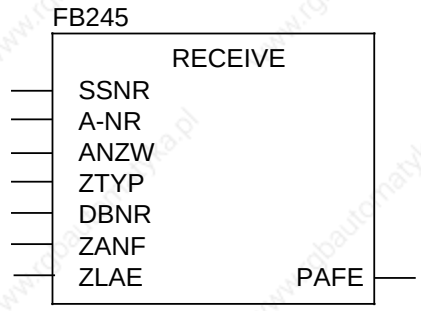
If "RW" is entered in the QTYP parameter, the block transfers the indirectly specified source and destination parameters to the CP. Then the destination parameters are sent along with the useful data (requested via the SEND ALL function) to the communications partner (WRITE function).

**The RECEIVE Block - FB245 -**

FB245 requests reception of data from a module with page addressing. A distinction is made between two functions modes:

- RECEIVE All
  Data can be received for any job. This function block substitutes for direct memory access.
- RECEIVE Direct
  Data is received for a specific job.

**Calling Function Block FB245**

| STL | | | | CSF/LAD |
|---|---|---|---|---|
| | : JC | FB | 245 |  |
| NAME | : RECEIVE | | | |
| SSNR | : | KY | 0,10 | |
| A-NR | : | KY | 0,101 | |
| ANZW | : | FW | 24 | |
| ZTYP | : | KS | DB | |
| DBNR | : | KY | 0,10 | |
| ZANF | : | KF | +100 | |
| ZLAE | : | KF | - 1 | |
| PAFE | : | FY | 23 | |
| | : *** | | | |



**Description of the RECEIVE ALL Function**

The block needs the following parameters in RECEIVE ALL mode:

- SSNR - interface number
- A-NR - job number (assign "0")
- ANZW - job status word
- PAFE - parameter assignment error byte

All other parameters are irrelevant for this job.

The CP provides the following information via the communications area:

- address of the job status word
- type of data
- amount of data
- start address of the data area

The following bits are evaluated or set/reset in the status word for the pertinent job:

- data transfer disabled
- data transfer completed
- data transfer in progress

The block enters the amount of data transferred for a job in the data word that follows the job status word.

The RECEIVE block must be called in the control program in "ALL" mode at least once per interface when
- the CP wants to give data to the PLC on its own initiative.
- the amount of data to be received with RECEIVE DIRECT exceeds the specified frame size.
- the CP uses RECEIVE DIRECT only to enable receive data, and transfers data to the PLC via "background communications".

You can call FB245 in RECEIVE ALL mode in
- the cyclic program (e.g., in OB1)
- the service routine for timed interrupts (e.g. prompter block)
- the service routine for process interrupts

**Description of the RECEIVE DIRECT Function**

The RECEIVE DIRECT function works with the following parameters:
- SSNR - interface number
- A-NR - job number (assign " 0")
- ANZW - job status word
- PAFE - parameter assignment error byte
- ZTYP - destination type
- DBNR - data block number
- ZANF - destination start address
- ZLAE - amount of destination data

Normally, the RECEIVE DIRECT function is called in the cyclic part of the control program. This block can also be called in an interrupt service routine, but the job status word is not updated cyclically in this case. The CONTROL block must then perform this task.

The RECEIVE block communicates with the CP on a handshaking basis under the following conditions only:
- RLO "1" has been forwarded to the function block.
- The CP has enabled the job. (The "RECEIVE ready" bit in the job status word is set).

When RLO "0" is forwarded, only the job status word is updated.

If "NN" is assigned to the ZTYP parameter, the CP must provide the destination parameters. Otherwise, the job is aborted with an error.

If the CP provides the destination parameters, when ZTYP is not "NN", only the parameter specifications in the block are noted.

Large amounts of data can be received in the form of frames only. Only one data frame can be received at a time with RECEIVE DIRECT. The remaining data or additional data frames must therefore be transferred to the PLC with RECEIVE ALL.

**The FETCH Block - FB246 -**

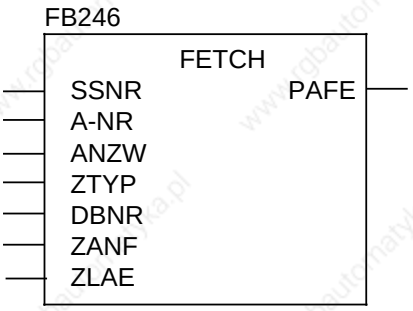
FB246 requests that data can be fetched from a communications partner over a CP. The data is received via function block FB245 in RECEIVE ALL mode. You can use the FETCH block only to fetch data for a specific job (FETCH DIRECT function).

**Calling the FETCH Block (Example)**

**STL**

```
      : JU    FB    246
NAME  : FETCH
SSNR  :       KY    0,10
A-NR  :       KY    0,101
ANZW  :       FW    9
ZTYP  :       KS    DB
DBNR  :       KY    0,46
ZANF  :       KF    + 5
ZLAE  :       KF    + 20
PAFE  :       FY    14
      : ***
```

**CSF/LAD**




**Description of the FETCH Function**

All parameters must be assigned for the FETCH function. The destination parameters (ANZW, ZTYP, DBNR, ZANF, ZLAE) are passed to the CP during handshaking. As soon as the requested data arrives, the CP provides the RECEIVE ALL block with both parameters and data. The FETCH block itself does not transfer or receive data.

The FETCH job is activated under the following conditions:
- RLO "1" has been forwarded to the function block.
- The CP has enabled the function. (The "SEND/FETCH in progress" bit is "0").

If "RW" is assigned to the ZTYP parameter, the FETCH block transfers the source and destination parameters and the address of the job status word to the CP.

FETCH can be invoked in the cyclic program or in an interrupt service routine. The FETCH or CONTROL block updates the job status word.

**The CONTROL Block - FB247 -**

FB247 updates the job status word for a specific job or indicates which job is currently in progress.

**Calling FB247 (Example)**

| STL | | | | CSF/LAD |
|---|---|---|---|---|
| | : JU | FB | 247 | FB247 |
| NAME | : CONTROL | | | CONTROL |
| SSNR | : | KY | 0,10 | SSNR |
| A-NR | : | KY | 0,101 | A-NR |
| ANZW | : | FW | 20 | ANZW PAFE |
| PAFE | : | FY | 22 | |
| | : *** | | | |

**Description of the CONTROL Function**

The CONTROL function requires the following parameters:
- SSNR - interface number
- A-NR - number of the job to be monitored
- ANZW - job status word where the result is to be stored
- PAFE - parameter assignment error byte

The CONTROL block implements different functions depending on the job number.

**A-NR="0"**

The CP is asked which job is currently in progress. The CP writes the number of the current job in job location 0. The contents of this location are transferred to the low-order byte of the job status word when the CONTROL block is processed.

**A-NR "0"**

The block executes in CONTROL DIRECT mode:
- The status of a specific job is interrogated.
- The job status word is updated.

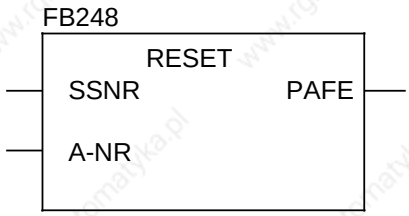
Processing of this block does not depend on the RLO. However, FB247 should be called in the cyclic part of the control program.

**The RESET Block - FB248 -**

FB248 resets a job executing over the specified interface. RESET can execute in two different modes:

- RESET All
  If you assign "0" as the job number, all jobs for the specified interface are reset.
- RESET Direct
  If you assign a number " 0" as the job number, only the specified job is reset.

**Calling FB248 (Example)**

| STL | CSF/LAD |
|---|---|
| : JU FB 248<br>NAME : RESET<br>SSNR : KY 0,1<br>A-NR : KY 0,0<br>PAFE : FY 111<br>: *** |  |

**Parameter Description**

FB248 requires the following parameters:

- SSNR - interface number
- A-NR - number of the job that is to be reset
- PAFE - parameter assignment error byte

**RESET Function Description**

In both modes,

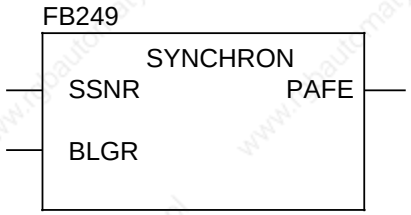
- the job data are deleted.
- active jobs are aborted.

FB248 executes dependent on the RLO, and can be invoked in both the cyclic program and in an interrupt service routine.

**The SYNCHRON Block- FB249 -**

Each time the PLC is restarted, FB249 initializes the interface on a module with page addressing for communication with the control program. This synchronization is essential for proper execution of the data handling blocks.

**Calling FB249 (Example)**

| STL | CSF/LAD |
|---|---|
| : JU FB 249<br>NAME : SYNCHRON<br>SSNR : KY 0,1<br>BLGR : KY 0,5<br>PAFE : FY 100<br>: *** |  |

**Parameter Description**

FB249 requires the following parameters:
- SSNR - interface number
- BLGR - frame size
- PAFE - parameter assignment error byte

**SYNCHRON Function Description**

After you enter the desired frame size for the BLGR parameter, the CP checks this value according to module-specific criteria and determines the final frame size.

In certain cases, this means that the frame size specified in the parameter is invalid.

The final frame size specifies how much data (bytes) can be transferred directly when the SEND and RECEIVE blocks are called. For larger amounts of data, continuation frames are generated and transferred with the ALL functions of these blocks.

FB249 synchronizes the PLC and the CP on each PLC restart. Consequently, FB249 should be called in RESTART blocks OB21 and OB22. FB249 executes when it receives RLO "1".

## 11.1.4 The Integral "COMPR" Block

The integral "COMPR" block (no. 238) compresses the internal program memory. If you want to use integral FB "COMPR" with block number 238, you must not have assigned number 238 to any other FB. If you nevertheless want to use a user-written block with number 238 (and ***not*** the integral FB238), proceed as follows:

POWER ON
Overall Reset
Transfer "user" FB with number 238 to the PLC
Set mode selector to RUN

or

Plug E(E)PROM with "user" FB (number 238) into the submodule
POWER ON
Overall Reset
Set mode selector to RUN.

### Calling the Function Block

The "PLC Compress" function is initiated by invoking FB238 in the application program. This function block returns the "AKT" bit, which indicates whether the "Compress" function is or is not still in progress. The "ERR" bit is set if the function cannot be executed.

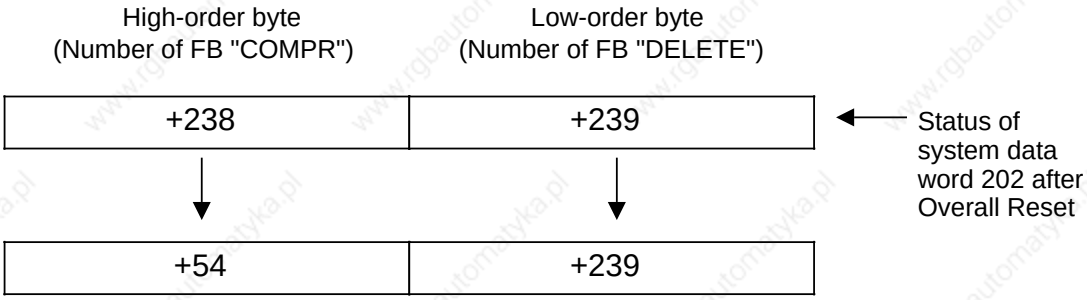
| STL | CSF/LAD |
|---|---|
| : A I 0.0<br>: AN F 0.0<br>: = F 0.1<br>: A I 0.0<br>: = F 0.0<br>:<br>: A F 0.1<br>: JC FB 238<br>NAME : COMPR<br>AKT : F 1.0<br>ERR : F 1.1 | FB238<br>COMPR<br>AKT<br>ERR |

**Note**

The FB COMPR has the same effect as the programmer "Compress" function, i.e. if FB COMPR is active, other programmer/OP functions will be rejected, e.g. STATUS or block input/output.
Generation and deletion of a DB with G DB in this case results in CPU Stop (TRAF).

It is also possible to renumber FB238
- in DB1 (see Section 11.3) or
- By changing system data word 202 in the RESTART OB (OB21 or OB22) using the "T RS 202" operation. System data word 202 must ***not*** be changed using the "DISPL. ADDR.", "TNB", "TIR" or "TDI" operations!

**Example:**
The integral FB238 (COMPR) is to receive the number 54. Make sure that FB number 54 has not already been assigned to another FB (if a user program FB has the same number as the integral FB "COMPR", the "old" FB number is retained in system data word 202). The number of integral FB239, which can also be changed by system data word 202, should be retained!




The STL program (programmed here in FB3) takes the following form:

| FB3 STL | Explanation |
|---|---|
| :<br>L KY 54,239<br>T RS 202<br>L RS 202<br>!=F<br>BEC<br>STS<br>BE | FB3 is called by OB21 and OB22.<br>The numbers of FBs COMPR and DELETE are transferred to system data word 202.<br>Read RS 202 (check if new number has been accepted)<br><br>If not ... STOP! |

## 11.1.5 Integral FB "DELETE"

The integral FB "DELETE" (No. 239) deletes a block. If you want to use integral FB "DELETE" with block number 239, you must not have assigned number 239 to any other FB. If you nevertheless want to use a user-written block with number 239 (and ***not*** the integral FB239), proceed as follows:

POWER ON
Overall Reset
Transfer "user" FB with number 239 to the PLC
Set mode selector to RUN

or

Plug E(E)PROM with "user" FB (number 239) into the submodule receptacle
POWER ON
Overall Reset
Set mode selector to RUN.

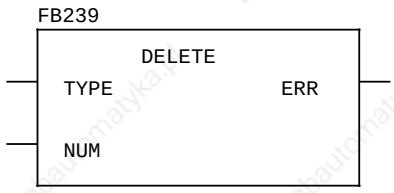
**Initializing the Integral FB DELETE**

Initialize the integral FB239 as follows:

- Store the type of the block to be deleted in an input word, flag word or data word as an ASCII character (KS). The characters OB. PB, FB, SB and DB are permissible as block identifiers.
- Store the block number in an input byte or a flag byte.

You must also specify a flag byte or an output byte which the operating system can use to flag errors (see Table 11-9).

**Calling the Function Block (Example)**

| STL | CSF/LAD |
|---|---|
| `      : JU   FB  239`<br>`NAME : DELETE`<br>`TYPE : FW   5`<br>`NUM  : FY   7`<br>`ERR  : FY   8`<br>`      : ***` |  |

Contents of FW 5: Block type in ASCII code (e.g. PB for program block)
Contents of FY 7: Block number (e.g. KF+7)
Contents of FY 8: No entry is made in FY 8 until the function block has been invoked (see Table 11-9)

**Hinweis**

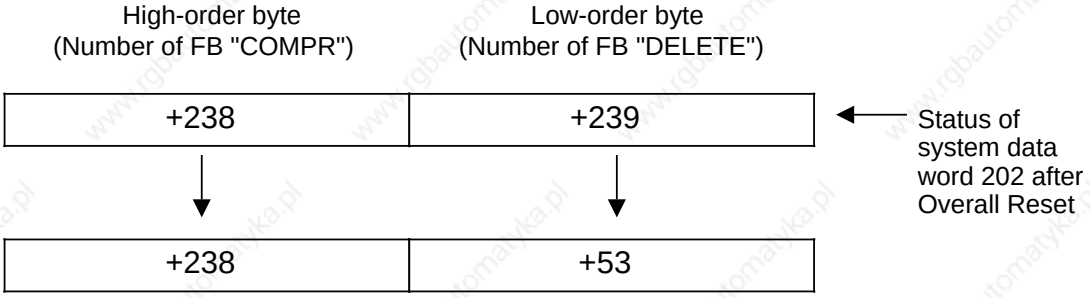
If Delete is currently active, programmer/OP functions can be rejected.

**Table 11-9. Error Bits Set by FB239 (ERR Parameter)**

| Hexadecimal Value of the ERR Parameter | Description |
|---|---|
| 00 | No error |
| F0 | No such block |
| F1 | Invalid block type specified in the TYPE parameter |
| F2 | Block exists, but has an EPROM identifier |
| F4 | DELETE function cannot execute because another function is in progress (e.g. a programmer function) |

It is also possible to renumber FB239 (DELETE)
- in DB1 (see Section 11.3) or
- By changing system data word 202 in the RESTART OB (OB21 or OB22) using the "T RS 202" operation. System data word 202 must ***not*** be changed using the "DISPL. ADDR.", "TNB", "TIR" or "TDI" operations!

**Example:**
The integral FB239 (DELETE) is to receive the number 53. Make sure that FB number 53 has not already been assigned to another FB (if a user program FB has the same number as the integral FB "DELETE", the "old" FB number is retained in system data word 202).
The number of integral FB238, which can also be changed by system data word 202, should be retained!




The STL program (programmed here in FB4) takes the following form:

| FB4 STL | Explanation |
|---|---|
| :<br>L KY 238,53<br>T RS 202<br>L RS 202<br>!=F<br>BEB<br>STS<br>BE | FB4 is called by OB21 and OB22.<br>The numbers of FBs COMPR and DELETE are transferred to system data word 202.<br>Read RS 202 (check if new number has been accepted)<br><br>If not ... STOP! |

## 11.2 Organization Blocks

Besides function blocks, organization blocks are also integrated in the CPUs of the S5-115U programmable controller.

### 11.2.1 OB31 Scan Time Triggering

A scan time monitor monitors the program scan time. If program scanning takes longer than the specified scan monitoring time (e.g., 500 msec.), the CPU enters the "STOP" mode.

This situation can occur, for instance, when
- The control program is too long.
- The program enters a continuous loop.

The scan time monitor can be started at any point in the control program using the OB31 call (JU OB31), i.e. the scan monitoring time is restarted.

Prerequisite: SYSTEM OPERATIONS enabled on the programmer.

The scan monitoring time can be set in the following ways:
- In system data word 96 ($EAC0_H$) (see Chapter 2)
- In DB1 (see Section 11.3).

### 11.2.2 OB160 Variable Time Loop

OB160 simulates operation execution times. This makes you independent of the different operation execution times of the various CPUs and you can program wait times uniformly for all CPUs of the S5-115U range.

Proceed as follows:
The waiting time must be loaded into the ACCUM in µsec. (range: 120 to 65535 or $78_H$ to $FFFF_H$).

**Example:**
A waiting time of one millisecond is to be programmed.

```
L  KF +1000
JU OB 160.
```

Please note the following when programming OB160:
A process interrupt (OB2 to OB5) and the timed interrupt (OB6) can interrupt the waiting time (provided no interrupt disable (IA) has been programmed). The waiting time stops during the interrupt! Similarly, the waiting time is increased by running PG/OP operations. The times set are therefore minimum times!
OBs 10 to 13 cannot interrupt OB160!

*In the case of CPU 944: 190 µs

### 11.2.3 OB251 PID Control Algorithm

The operating systems of the central processing units have an integral PID control algorithm which you can use for your own purposes with the help of organization block OB251.
Before calling OB251, a data block (PID controller DB) containing the controller parameters and other controller-specific data must be opened. The PID control algorithm is called periodically (sampling interval) and generates the manipulated variable. The more closely the sampling interval is observed, the more accurately can the controller fulfill its appointed task. The control parameters specified in the controller DB must be matched to the sampling interval. Typically, timed interrupts are serviced by a time block (OB10 to OB13).
Timed-interrupt OBs can be called at intervals between 10 msec. and 10 minutes. The maximum execution time of the PID control algorithm is 2 msec.

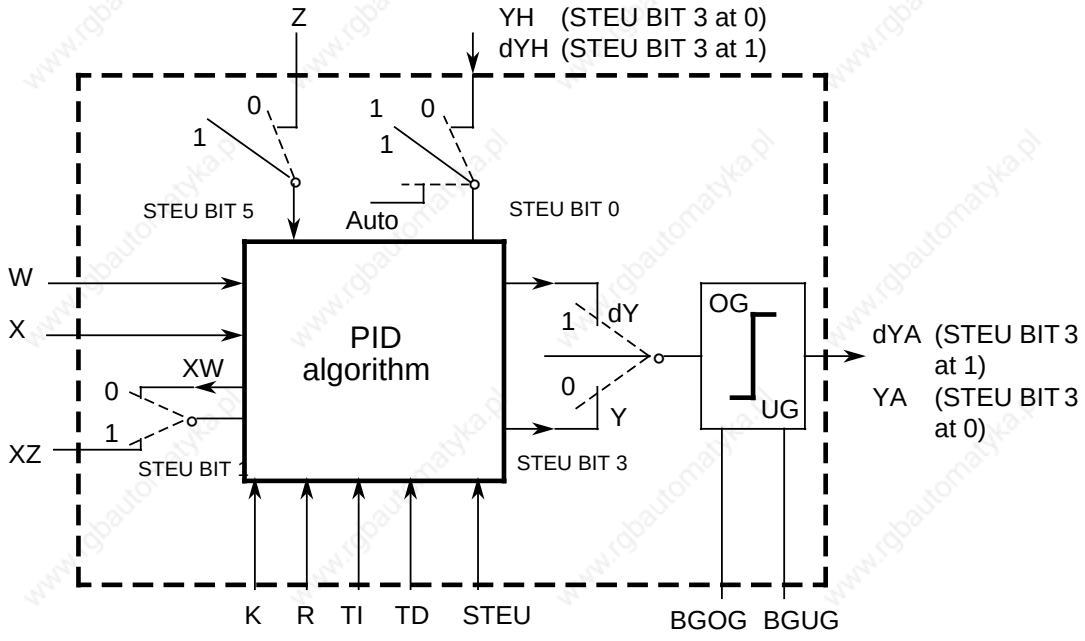



**Figure 11-3. Block Diagram of the PID Controller**

Legend:

| | | | | | |
|---|---|---|---|---|---|
| K | = | Proportional coefficient<br>K>0 positive control direction<br>K<0 negative control direction | Y<br>dY<br>YH | =<br>=<br>= | Manipulated variable<br>Correcting increment<br>Value for manual input |
| R | = | R parameter (usually 1000) | dYH | = | Correcting increment for manual input |
| TA | = | Sampling interval | BGOG | = | Upper limiting value |
| TN | = | Integral-action time | BGUG | = | Lower limiting value |
| TV | = | Derivative-action time | X | = | Actual value |
| TI | = | TA/TN | Z | = | Disturbance variable |
| TD | = | TV/TA | XZ | = | Substitute variable for control deviation |
| STEU | = | Control word | YA | = | Controller output: manipulated variable limited |
| W | = | Setpoint | dYA | = | Controller output: correcting increment limited |
| XW | = | Controller difference | | | |

The continuous action controller is designed for controlled systems such as those used in pressure, temperature, or flow rate control.

The "R" parameter sets the proportional 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 parameters (R, TI, and TD) by presetting the pertinent data words with zero. This enables you to implement all required controller structures without difficulty, e.g., PI, PD, or PID controllers.

You can forward the system 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 by setting the associated bits in control word STEU when the D controller is initialized.

**Table 11-10. Description of the Control Bits in Control Word STEU**

| Control Bit | Name | Signal State | Description |
|---|---|---|---|
| 0 | AUTO | 0 | Manual mode<br>The following variables are updated in Manual mode:<br>1) $X_K$, $XW_{K-1}$ and $PW_{K-1}$<br>2) $XZ_K$, $XZ_{K-1}$ and $PZ_{K-1}$, when STEU bit 1=1<br>3) $Z_K$ and $Z_{K-1}$, when STEU bit 5=0<br>Variable $dD_{K-1}$ is set to 0. The algorithm is not computed. |
| | | 1 | Automatic mode |
| 1 | XZ EIN | 0 | $XW_k$ is forwarded to the differentiator. The XZ input is ignored. |
| | | 1 | A variable other than $XW_k$ is forwarded to the differentiator. |
| 2 | REG AUS | 0 | Normal controller processing |
| | | 1 | When the controller is invoked (OB251), all variables (DW 18 to DW 48 ) with the exception of K, R, TI, TD, BGOG, BGUG, $YH_k$ and $W_k$ are reset in the controller DB. The controller is deactivated. |
| 3 | GESCHW | 0 | Correction algorithm |
| | | 1 | Velocity algorithm |
| 4 | HANDART | 0 | When GESCHW=0:<br>Following the transfer to Manual mode, the specified manipulated variable value YA is adjusted exponentially to the manual value in four sampling steps. Additional manual values are then forwarded immediately to the controller output.<br>When GESCHW=1:<br>The manual values are forwarded immediatley to the controller output.<br>The limiting values are in force in Manual mode. |
| | | 1 | When GESCHW=0:<br>The manipulated variable last output is retained.<br>When GESCHW=1:<br>Correction increment $dY_K$ is set to zero. |
| 5 | NO Z | 0 | With feedforward control |
| | | 1 | No feedforward control |
| 6 to 15 | - | | The PID algorithm uses these bits as auxiliary flags. |

The control program can be supplied 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 modification (correction rate algorithm).

**Correction Rate Algorithm**

The relevant correction increment $dY_k$ is computed at instant t= k · TA according to the following formula:

- Without feedforward control (D11.5=1); XW is forwarded to the differentiator (D11.1=0)

$dY_k = K[(XW_k - XW_{k-1}) R+TI \cdot XW_k+ (TD (XW_k - 2XW_{k-1} + XW_{k-2}) + dD_{k-1})]$

$= K (dPW_k + dI_k + dD_k)$

- With feedforward control (D11.5=0); XW is forwarded to the differentiator (D11.1=0)

$dY_k = K[(XW_k - XW_{k-1}) R+TI \cdot XW_k+ (TD (XW_k - 2XW_{k-1} + XW_{k-2}) + dD_{k-1})]+(Z_k-Z_{k-1})$

$= K (dPW_k + dI_k + dD_k)+dZ_k$

- Without feedforward control (D11.5=1); XZ is forwarded to the differentiator (D11.1=1)

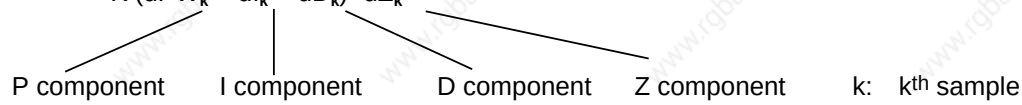
$dY_k = K[(XW_k - XW_{k-1}) R+TI \cdot XW_k+ (TD (XZ_k - 2XZ_{k-1} + XZ_{k-2}) + dD_{k-1})]$

$= K (dPW_k + dI_k + dD_k)$

- With feedforward control (D11.5=0); XZ is forwarded to the differentiator (D11.1=1)

$dY_k = K[(XW_k - XW_{k-1}) R+TI \cdot XW_k+ (TD (XZ_k - 2XZ_{k-1} + XZ_{k-2}) + dD_{k-1})]+(Z_k-Z_{k-1})$

$= K (dPW_k + dI_k + dD_k)+dZ_k$




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:

$dPW_k = (XW_k- XW_{k-1})R$

$dI_k = TI \cdot XW_k$

$dD_k = (TD \cdot QW_k+dD_{k-1})$ when XW is applied

$= (TD \cdot QZ_k+dD_{k-1})$ when XZ is applied

$dZ_k = Z_k - Z_{k-1}$

**Correction Algorithm**

The formula used to compute the correction rate algorithm is also used to compute the correction algorithm.
In contrast to the correction rate algorithm, however, the sum of all correction increments computed (in DW 48), rather than the correction increment $dY_k$ is output at sampling instant $t_k$.

At instant $t_k$ , manipulated variable $Y_k$ is computed as follows:

$$Y_k = \sum_{m=0}^{m=k} dY_m$$

**Initializing the PID Algorithm**

OB251's interface to its environment is the controller DB.
All data needed to compute the next manipulated variable value is stored in this DB. Each controller must have its own controller data block.
The controller-specific data are initialized in a data block that must comprise at least 49 data words.
The CPU goes to STOP with a transfer error (TRAF) if no DB has been opened or if the DB is too short.

**Caution**

Make sure that the right controller DB has been invoked before calling control algorithm OB251.

**Table 11-11. Format of the Controller DB**

| Data Word | Name | Comments |
|---|---|---|
| 1 | K | Proportional coefficient (- 32 768 to+32 767) for controllers without D component<br>Proportional coefficient (- 1500 to+1500) for controllers with D component[1]<br>K is greater than zero when the control direction is positive, and less than zero when the control direction is negative; the specified value is multiplied with the factor 0.001. |
| 3 | R | R parameter (- 32 768 to+32 767) for controllers without D component<br>R parameter (-1500 to+1500) for controllers with D component[1]<br>Normally 1 for controllers with P component: the specified value is multiplied with the factor 0.001. |
| 5 | TI | Constant TI (0 to 9999)<br>$TI = \frac{\text{Sampling interval TA}}{\text{Integral-action time TN}}$<br>The specified value is multiplied with a factor of 0.001. |
| 7 | TD | Constant TD (0 to 999)<br>$TD = \frac{\text{Derivative-action time TV}}{\text{Sampling interval TA}}$ |
| 9 | W | Setpoint (- 2047 to +2047) |
| 11 | STEU | Control word (bit pattern) |
| 12 | YH | Value for manual operation (- 2047 to +2047) |
| 14 | BGOG | Upper limiting value (- 2047 to +2047) |
| 16 | BGUG | Lower limiting value (- 2047 to +2047) |

1 Larger gains are possible if abrupt changes to the system deviation are sufficiently small. Large changes of the system deviation should therefore be divided up into small changes; e.g. by feeding the setpoint via a ramp function.
The factor 0.001 is an approximate value. The precise value for the factor is 1/1024 or 0.000976.

**Table 11-11. Format of the Transfer Block (Continued)**

| Data Word | Name | Comments |
|---|---|---|
| 22 | X | Actual value (- 2047 to+2047) |
| 24 | Z | Disturbance variable (- 2047 to+2047) |
| 29 | XZ | Derivative time (- 2047 to+2047) |
| 48 | YA | Output variable (- 2047 to+2047) |

All parameters (with the exception of the control word STEU) must be specified as 16-bit fixed point numbers.

> **Note**
>
> The PID algorithm uses the data words that are not listed in the table as auxiliary flags.

**Initializing and Invoking the PID Controller in the STEP 5 Program**

A number of different PID controllers can be implemented by calling OB251 repeatedly, so make sure that the relevant controller DB has been invoked before calling OB251.

> **Note**
>
> Important controller data are stored in the high-order byte of control word DW 11 (DL 11). Make sure that only T DR 11/SU D 11.0 to D 11.7 or RU D 11.0 to D 11.7 operations are used to modify user-specific bits in the control word.

**Selecting the Sampling Interval**

The value selected as sampling interval must not be excessiveley high in order to be able to use the well-known analog method in the case of digital control loops.
Experience has shown that a sampling interval of approximately 1/10 of the counter constant $T_{RKdom}$* produces a control result comparable to the equivalent analog result. Dominant system time constant$T_{RKdom}$ determines the step response of the closed control loop.

$$T_A = 1/10 \cdot T_{RKdom}$$

In order to ensure the constancy of the sampling interval, OB251 must always be invoked in the service routine for timed interrupts (OB13).

* $T_{RKdom}$=Dominant system time constant of the closed control loop

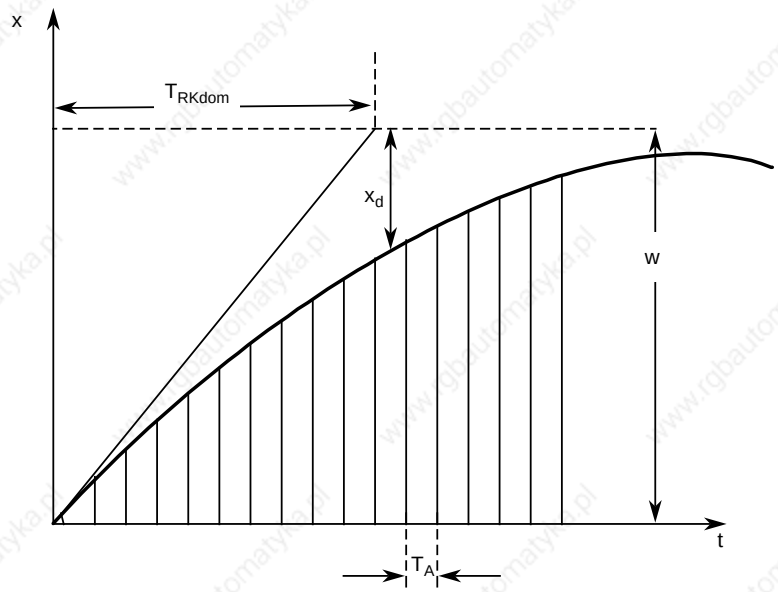



**Figure 11-4. Estimating the Dominant System Time Constant ($T_{RKdom}$)**

**Example for the Use of the PID Control Algorithm**

Using a PID controller to keep an annealing furnace at a constant temperature.
The temperature setpoint is entered via a potentiometer.
The setpoints and actual values are acquired using an analog input module and forwarded to the controller. The computed manipulated variable is then output via an analog output module.
The controller mode is set in input byte 0 (see control word DW 11 in the controller DB).
You must use the well-known controller design procedure to determine how to tune the controller for each controlled system.

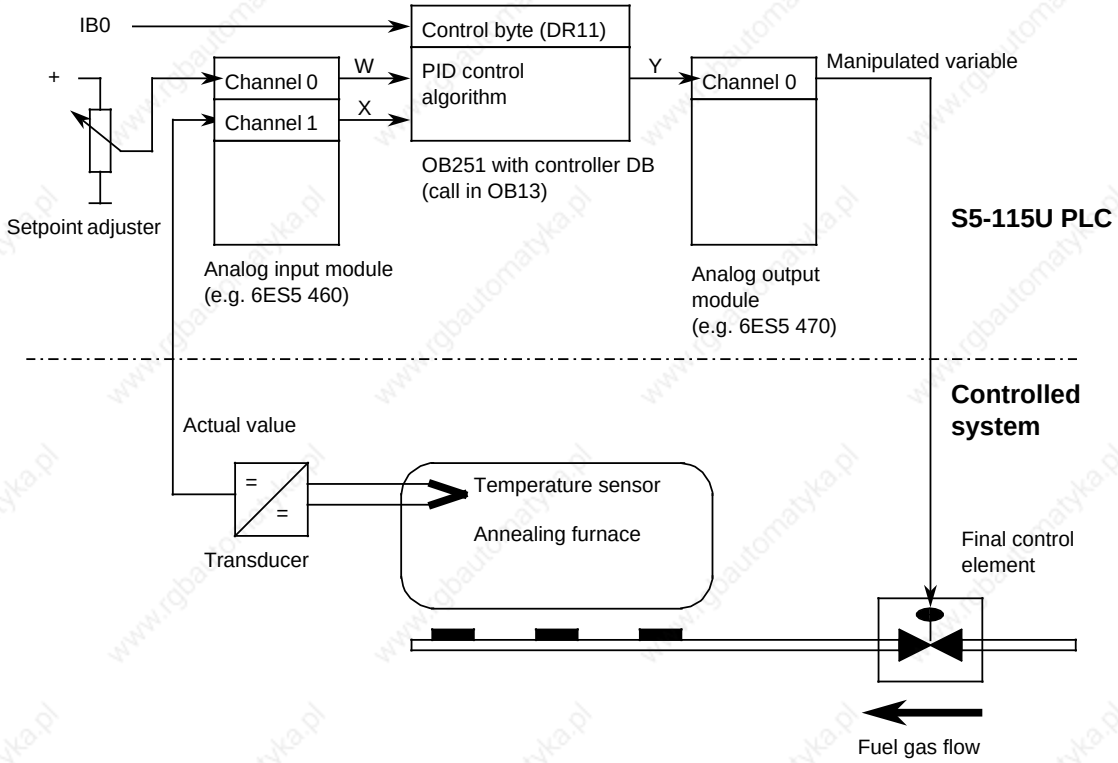



**Figure 11-5. Process Schematic**

The analog signals of the setpoint and actual values are converted into corresponding digital values in each sampling interval (set in OB13). OB251 uses these values to compute the new digital manipulated variable, from which, in turn, the analog output module generates a corresponding analog signal. This signal is then forwarded to the controlled system.

Invoking the Controller in the Program:

| OB13 STL | Description |
|---|---|
| :<br>:JU FB 10<br>NAME :REGLER 1<br>:<br>:<br>:<br>:<br>:<br>:<br>:<br>:BE | PROCESS CONTROLLER<br><br>THE CONTROLLER'S SAMPLING INTERVAL<br>DEPENDS ON THE TIME BASE USED<br>TO CALL OB 13 (SET IN SD 97).<br>THE DECODING TIME OF THE<br>ANALOG INPUT MODULES MUST BE TAKEN<br>INTO ACCOUNT WHEN SELECTING<br>THE SAMPLING INTERVAL. |

| FB10 STL | Description |
|---|---|
| NAME :REGLER 1 | |
| : | |
| :C DB 30 | SELECT CONTROLLER'S DB |
| : | |
| : | ********** |
| : | READ CONTROLLER'S CONTROL BITS |
| : | ********** |
| : | |
| :L PY 0 | READ CONTROLLER'S |
| :T FY 10 | CONTROL INPUTS |
| :T DR 11 | AND STORE IN DR11 |
| : | CAUTION: |
| : | DL11 CONTAINS IMPORTANT CONTROL |
| : | DATA FOR OB 251 |
| : | THE CONTROL BIT MUST |
| : | THEREFORE BE TRANSFERRED WITH |
| : | T DR11 TO PREVENT |
| : | CORRUPTING DL11 |
| : | |
| : | ********** |
| : | READ ACTUAL VALUE AND SETPOINT |
| : | ********** |
| : | |
| :A F 12.0 | FLAG 0 (FOR UNUSED FUNCTIONS |
| :R F 12.0 | IN FB250) |
| :AN F 12.1 | FLAG 1 |
| :S F 12.1 | |
| : | |
| :JU FB 250 | READ ACTUAL VALUE |
| NAME :RLG:AE | |
| BG : KF +128 | MODULE ADDRESS |
| KNKT : KY 0,6 | CHANNEL NO. 0, FIXED-POINT BIPOLAR |
| OGR : +2047 | UPPER LIMIT FOR ACTUAL VALUE |
| UGR : -2047 | LOWER LIMIT FOR ACTUAL VALUE |
| EINZ : F 12.0 | NO SELECTIVE SAMPLING |
| XA : DW 22 | STORE SCALED ACTUAL VAL. IN CONTR. DB |
| FB : F 12.2 | ERROR BIT |
| BU : F 12.3 | RANGE VIOLATION |
| TBIT : F 12.4 | ACTIVITY BIT |
| : | |
| : | |
| :JU FY 250 | READ SETPOINT |

| FB10 (Continued) STL | Description |
|---|---|
| NAME :RLG:AE | |
| BG : KF +128 | MODULE ADDRESS |
| KNKT : KY 1,6 | CHANNEL NO. 1, FIXED-POINT BIPOLAR |
| OGR : KF +2047 | UPPER LIMIT FOR SETPOINT |
| UGR : KF -2047 | LOWER LIMIT FOR SETPOINT |
| EINZ : F 12.0 | NO SELECTIVE SAMPLING |
| XA : DW 9 | STORE SCALED SETPOINT IN CONTR. DB |
| FB : F 13.1 | ERROR BIT |
| BU : F 13.2 | RANGE VIOLATION |
| TBIT : F 13.3 | ACTIVITY BIT |
| : | |
| :A F 10.0 | IN MANUAL MODE, THE SETPOINT IS |
| :JC =WEIT | SET TO THE ACT. VAL. TO FORCE |
| :L DW 22 | THE CONTROLLER TO REACT |
| :T DW 9 | TO A SYSTEM DEVIATION, IF ANY, |
| : | WITH A P STEP |
| : | ON TRANSFER |
| : | TO AUTOMATIC |
| : | MODE |
| WEIT : | |
| : | ********** |
| :JU OB 251 | INVOKE CONTROLLER |
| : | ********** |
| : | |
| : | ********** |
| : | OUTPUT MANIPULATED VARIABLE Y |
| : | ********** |
| :JU FB 251 | |
| NAME :RLG:AA | |
| XE : DW 48 | FORWARD MAN. VAR. TO ANALOG OUTPUT MOD. |
| BG : KF +176 | MODULE ADDRESS |
| KNKT : KY 0,1 | CHANNEL 0, FIXED-POINT BIPOLAR |
| OGR : KF +2047 | UPPER LIMIT FOR ACTUATING SIGNAL |
| UGR : KF -2047 | LOWER LIMIT FOR ACTUATING SIGNAL |
| FEH : F 13.5 | ERROR BIT WHEN LIMITING VALUES DEFINED |
| BU : F 13.6 | RANGE VIOLATION |
| :BE | |

| DB30 | STL | Description |
|---|---|---|
| 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; | SETPOINT 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 CONT. LIMIT BGOG, FACTOR 1 |
| 15: | KH = 0000; | (VALUE RANGE: -2047 TO 2047) |
| 16: | KF = -02000; | LOWER CONT. LIMIT 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; | FEEDFORWARD XZ 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: | | |

### 11.2.4 OB254 Read In Digital Input Modules (CPU 944 only)

OB254 (which can be invoked with JU OB254 or JC OB254) transfers the digital inputs to the process input image (PII). In contrast to cyclic updating of the PII, OB254 does not take bit 1 into account in system data word RS 120, i.e. the Enable bit for cyclic updating of the PII.

### 11.2.5 OB255 Transfer the Process Output Image (PIQ) to the Output Modules (CPU 944 only)

OB255 (which can be invoked with JU OB255 or JC OB255) forwards the process output image to the digital output modules without regard to bit 2 in system data word RS 120, i.e. the Enable bit for cyclic output of the PIQ to the digital output modules.

## 11.3 DB1: Initializing Internal Functions

The CPU has functions which you can set to your own requirement. For example, you can initialize the following:

- Integral hardware clock (in the case of CPU 943 and CPU 944 with two interfaces each)
- Data interchange over SINEC L1
- Call interval for time-controlled program execution (OB10 to 13)
- System characteristics (e.g. scan time monitoring)
- Address for parameter error code.

You can initialize these functions in data block DB1.

### 11.3.1 Configuration and Default Settings for DB1

To make it easier for you to assign parameters, data block 1 is already integrated in the programmable controller with preset values (default parameters). After performing an overall reset, you can load the default DB1 from the programmable controller into your programmer and display it on the screen:

```
 0:     KS  ='DB1 TFB: OB13 100 ; SDP:';
12:     KS  =' WD 500 ; END ';
19:
```

Parameter block identifiers

This default DB1 contains one ***parameter block*** each for the following functions:
- Specifying the call interval for OB13; parameter block "TFB:".
- System characteristics (scan time monitoring); parameter block "SDP:".

#### What Typifies a Parameter Block?

A parameter block contains all the parameters of one function; it always starts with a block identifier followed by a colon. The colon must be followed by at least one space. The semicolon (;) indicates the end of a parameter block. The ***parameters*** are contained between the block identifier and the semicolon (;).

### 11.3.2 Setting the Addresses for the Parameter Error Code in DB1 (An example of how to set the parameters correctly)

We recommend that you use this example when you start setting your parameters. The following two reasons explain why.

1. There are no default parameters in DB1 for parameter block "ERT:". You must therefore enter this block complete. We will explain the entries step by step. In doing so, you will quickly learn the rules for initializing.
2. Properly entered, parameter block "ERT:" makes it easy for you to correct parameter errors. For this reason, you should complete this block in DB1 before changing or entering other parameters.

To help find parameter errors easier and to help correct them, you can ask the programmable controller to output error messages in a coded form. All you have to do is to tell the programmable controller where it should store the error code. Make this input in parameter block "ERT:" of DB1.
The error code can be stored in:
- flag words
  or
- data words in a data block.

The entire error code consists of 20 flag bytes or 10 data words. You only need to indicate the start address for the error code in parameter block "ERT:".

**Procedure:**

Overall Reset has been performed on the CPU and the CPU is in the STOP state.

Display default DB1 on the programmer
Position the cursor on the E of the end identifier "END" at the end of default DB1
Now enter the shaded characters:

| DB1 | Explanation |
|---|---|
| 0: KS ='DB1 TFB: OB13 100 ; SDP:';<br>12: KS =' WD 500 ; ERT: ERR FW1 ;';<br>24: KS =' END';<br>26: | The parameter error code is stored from flag word FW 1 after cold restart (automatically after power restore or manually). |

Use the following check list to make sure your entries are correct.
- Is the block ID "ERT:" terminated by a colon?
- Is at least 1 filler (a blank space) added after the colon?
- Is the parameter name ("ERR") entered correctly?
- Does at least 1 filler (a blank space) follow the parameter name?
- Is the argument (for example "FW1") entered correctly?
- Does at least 1 filler (a blank space) follow the argument?
- Does a semicolon (;) indicate the block end?
- The end ID "END" concludes DB1

Transfer the changed DB1 to the programmable controller.
Switch the programmable controller from STOP to RUN.

Changed DB1 parameters are accepted.

If you did not store the parameter block "ERT:" in DB1, you can localize the error in the ISTACK if there was an incorrect parameter setting. However, you will not know what type of error is present. The same thing applies if you made an error when you input the parameter block "ERT:".

### 11.3.3 How to Assign Parameters in DB1

As illustrated in Section 11.3.2, you use the following steps to change or expand the preset values of DB1:

Display the default DB1, with its parameter block "ERT:" on the programmer.
Position the cursor on the desired parameter block.
Change or expand the parameters.
Transfer the changed DB1 to the programmable controller.
Switch the programmable controller from STOP to RUN.

Changed DB1 parameters are accepted.

The following applies when initializing in DB1:

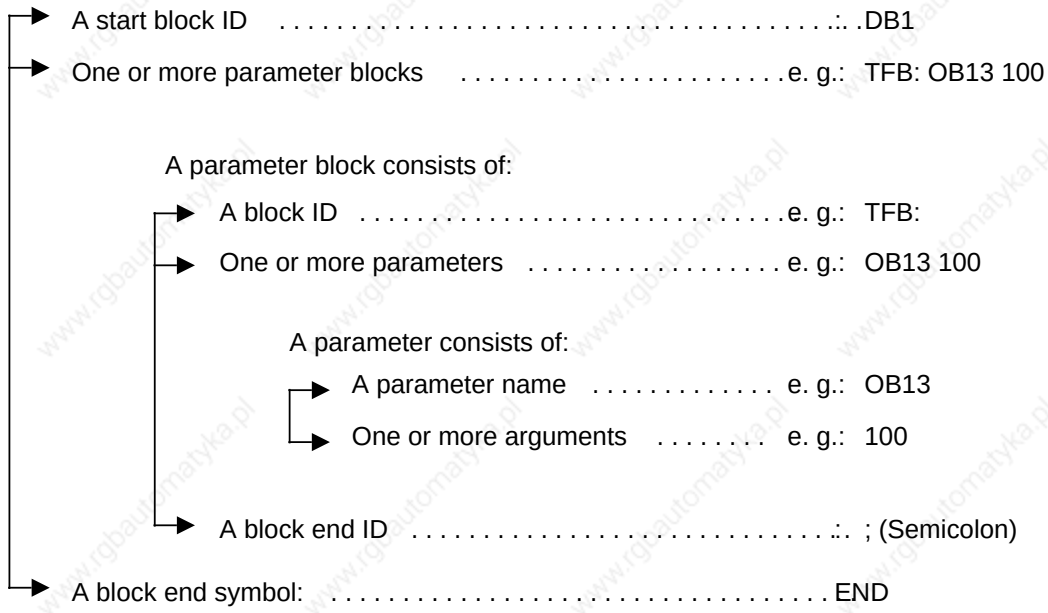
- ***Not*** all parameters of a parameter block must be defined in DB1. If some parameters are not defined in DB1, the default setting of the relevant system data word automatically applies!
- If you delete a DB1 in the PLC, the integral DB1 is retrieved by an overall reset of the CPU.
- If you define parameter blocks in DB1 which are irrelevant for the CPU (e.g. computer link for CPU 943), the CPU remains in the STOP state and generates a fault message (see Section 11.3.5).
- You can set parameters in either uppercase or lowercase.

**Note**

If the CPU detects a parameter error in DB1, it remains in the STOP mode even after switching from STOP  RUN (red STOP LED lights up).

### 11.3.4 Rules for Setting Parameters in DB1

DB1 consists of the following:




The following is a list of all the rules you must observe when changing parameters in DB1 or when completing whole parameter blocks. If you do not observe these rules, the CPU cannot interpret your entries. The structure of this DB1 depends on whether interprocessor communication flags must be defined or not!

1. **If interprocessor communication flags must be defined:**
   DB1 begins with the definition of the interprocessor communication flags as described in Section 12.1.1. The "DB1" start identifier for the other DB1 parameters follows the interprocessor communication flag end identifier ($EEEE_H$). The three characters must not be separated by spaces. The "DB1" start identifier must be followed by at least one space.
   **If no interprocessor communication flags must be defined:**
   DB1 begins with the "DB1" start identifier. The three characters must not be separated by spaces. The "DB1" start identifier must be followed by at least one space.

2. The start identifier (including space) is followed by the block identifier of a parameter block. The parameter blocks can follow any order in DB1. The block identifier indicates a block of related parameters. The block identifier "TFB" stands for "***T***imer ***F***unction ***B***lock" (time-controlled execution). The block identifier must be followed immediately by a colon (:). If the colon is missing, the CPU skips this block and outputs an error message. The block identifier and its colon must be followed by at least one space.

3. The parameter name comes next. Parameter names are names for single parameters within a parameter block. Within a block, the first four characters of a parameter name must be different from each other. After the parameter name, you must add at least one filler.

4. At least one argument is attached to each parameter name. An argument is either a number or a STEP-5 operand that you must enter. If several arguments belong to a parameter name, then every argument must be followed by at least one filler (even the last one).

5. Use a semicolon (;) to identify a block end. After the semicolon, you must enter at least one filler. Leaving out the semicolon leads to misinterpretation in the programmable controller.

6. After the semicolon, additional parameter blocks can follow. (Use steps 2 through 5 to create additional parameter blocks.)

7. After the end of the last parameter block, you must enter the end ID "END". This identifies the end of DB1. If you forget to enter an end ID, this leads to errors in the programmable controller.

Points 1 through 7 present the minimal requirements for setting the parameters. Beyond that, there are additional rules that make it easier for you to assign parameters.
For example:
- you have the ability to add comments
- you can expand the mnemonics used as parameter names in plain text.

Comments can be added anywhere a filler is allowed. The comment symbol is the pound (#) sign. The comment symbol must be placed at the beginning and at the end of your comment. The text between two comment symbols may not contain an additional #.
Example: #Comment# . At least one filler must follow the # sign.

In order to make it easier to read parameter names, you can add as many characters as you wish if you add an underscore (__) after the abbreviated parameter name.
Example: SF becomes SF__SENDMAILBOX .
At the end of the input, you must add at least one filler.

There is a rule of thumb that will help you to check DB1. You should include at least one filler in the following instances:
- ***after*** the start ID
- ***before and after*** the block ID, parameter name, argument, and semicolon

### 11.3.5 How to Recognize and Correct Parameter Errors

Should an error occur while assigning parameters and the programmable controller does not go to the RUN mode, you have two possibilities for recognizing errors:
- by using a parameter error code
- by using the analysis function "ISTACK"

Both possibilities are described below.

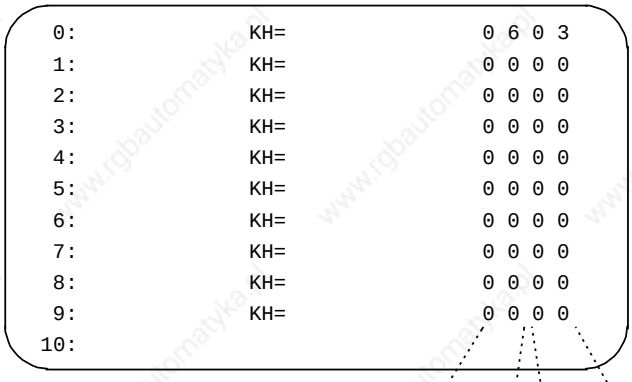
**Scanning the Parameter Error Code**

If you have entered a start address for the parameter error code in parameter block "ERT:" of DB1, then you can retrieve the cause of the error, and the error location information at this address.

The entire error code occupies 10 data words or 20 flag bytes. In the following examples and tables, we assume that the error code is stored in a data block starting with data word 0. The error code occupies DW 0 through DW 9. In the "Flag" operand area, this corresponds to Flag Byte 0 through Flag Byte 19.

> **Note**
>
> In the case of CPU 941/942, the error code area must not be a DB stored on an EPROM. In purely EPROM mode, use a flag area or previously tested error-free DB blocks.

**Example:**
You entered the start address DB3 DW 0 in parameter block "ERT:". The parameters set in DB1 have already been transferred to the programmable controller. Then you continue to set parameters in DB1. While attempting to transfer the changed DB1 parameters to the programmable controller, you find out that the programmable controller remains in the STOP mode. You suspect that the reason the programmable controller remains in the STOP mode is that there is a parameter error. To find the error, display DB3 on the programmer. The entire contents of DB3 appear on the screen. DW 0 through DW 9 contain the code for the parameter error. In the following figure, you see how your screen could look. Below the screen display, is a complete list of parameter error codes and their meanings.




Screen display with parameter error codes

| Cause of fault (which fault has occurred?) | DWL | | DWR | Fault Location (in which parameter block has the fault occurred?) |
|---|---|---|---|---|
| No fault | 00 | | 03 | SL1: SINEC-L1 |
| Start identifier (DB1) or end ident. (END) missing | 01 | | | |
| Unterminated comment before END or semicolon | 02 | | 06 | CLP:Clock parameters |
| missing before END or END identifier missing | | | 09 | TFB:Timer function block |
| | | | 10 | PFB:Placement of FB (238/239) |
| Block identifier syntax error or identifier unknown | 03 | | 11 | SDP:System characteristics |
| Parameter syntax error | 04 | | 99 | ERT:Error return |
| Argument syntax error or overrange | 05 | | F0 | Fault cannot be attributed to a block |
| | | | FF | Fault cannot be attributed to a block |
| Overrange in an argument | 06 | | | |
| Illegal parameter combination | 07 | | | |
| Not defined | 08 | | | |
| Not defined | 09 | | | |
| DB does not exist | 10 | | | |
| Insufficient space in DB (e.g. in clock DB) | 11 | | | |
| Error in specifying time | 12 | | | |
| in the date | 13 | | | |
| Error in time format | 14 | | | |
| Error in hour format | 15 | | | |

**Figure 11-6. Initialization Error Code and its Significance**

### Locating Parameter Errors in the ISTACK

If the PLC detects a parameter error in DB1 during restart, the PLC remains in the STOP state and stores the asolute (error) address as well as the relative (error) address in the ISTACK. The STEP address counter (SAC) in the ISTACK then points either

- **to** the address that contains the wrong input or directly
- **in front of** the address that contains the wrong input.

The addresses are byte addresses.

**Example:**
You have entered DB1 as follows; the shaded area represents an error.

```
 0:     KS  ='DB1 TFB: OB13 100 ; SDP:';
12:     KS  =' WD 3000 ; ERT: ERR MW1 ';
24:     KS  ='; END';
26:
```

The decimal number at the beginning of each input line is the word address of the first user-programmable character in that line. Each word consists of two characters (two bytes)

**Figure 11-7. DB1 with Parameter Error**

The ISTACK indicates the following as a result of this error:
- The absolute (error) address: $B14C_H$ (absolute SAC)
- The relative (error) address: $001C_H$ (relative SAC)

To locate the error accurately in your DB1, you must convert the relative byte address given as a hexadecimal number into a decimal word address.
The reason: The programmer counts the contents of a DB decimally and in words.
The SAC counts the contents of a DB hexadecimally and in bytes.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $001C_H$ | = | $28_D$ | $28_D$ | : | $2_D$ | = | $14_D$ |
| Hexadecimal byte address | | Decimal byte address | | | | | Decimal word address |

It follows that:
The error is in word address 14. In our example, address 14 (data word 14 and 15) is occupied by argument "3000". The entry "3000" is an error; reason: overrange.

## 11.3.6 Transferring the DB1 Parameters to the PLC

The CPU processes DB1 only after manual cold restart or after automatic cold restart after power restore.

You must perform a cold restart anytime you make changes to DB1. You can perform a cold restart by switching from:
- POWER OFF to POWER ON
  or from
- STOP to RUN

The programmable controller accepts the parameters from DB1 and stores them in the system data area.

**Note**

The programmable controller remains in the STOP mode if a parameter assignment error is found during start-up. The red LED lights up on the operator panel and ISTACK displays a DB1 addressing error.

### 11.3.7 Reference Table for Initializing DB1

| Parameter | Argument | Meaning |
|---|---|---|
| **Block Identifier: SL1:** | | **SINEC L1** |
| SLN | p | "***SL***ave ***N***umber" (p=1 to 30)<br>p=0 to 30 in the case of<br>CPU 943/944 with 2 interfaces) |
| SM | DBxDWy or FBy | Position of the ***S***end ***M***ailbox (start of SF) |
| RM | | Position of the ***R***eceive ***M***ailbox (start of EF) |
| CBR | | Position of the ***C***oordination ***B***yte ***R***eceive |
| CBS | | Position of the ***C***oordination ***B***yte ***S***end<br>(x=2 to 255; y=0 to 255) |
| PGN | p | ***PG*** bus ***N***umber (p=1 to 30)<br>Note: CBS and CBR are in a flag byte or in the high-order byte of the specified data word (DL)! |
| **Block Identifier:SDP:** | | **System-Dependent -Parameters** |
| WD | p | "***W***atch***D***og" (scan time monitoring)<br>can be set in milliseconds but only in steps of 10 msec. (p=0 to 2550) |
| RDLY | r | "***R***un***D***e***L***a***Y***" restart delay after POWER ON in msec. (r=0 to 65535)<br>Note: RDLY only becomes effective in the case of backup operation: in the case of EPROM operation, the setting is fixed at 1000 ms! |
| RT | Y/N | "***R***esident ***T***imers"(if "Y", all timers are retentive, if "N", only the first half are retentive) [1] |
| RC | Y/N | "***R***esident ***C***ounters" (if "Y", all counters are retentive, if "N", only the first half are retentive) [1] |
| RF | Y/N | "***R***esident ***F***lags" (if "Y", all flags are retentive, if "N", only the first half are retentive) [1] |
| PROT | Y/N | "***PROT***ection" activate software protection? (input/output of program no longer possible) |
| PIO | Y/N | "***P***rocess ***I***mage ***O***utput" disable output of process image? |
| PII | Y/N | "***P***rocess ***I***mage ***I***nput" disable read in of process image? |
| PRIO | s | OB6 ***PRIO***rity (the following is the descending order of priority:)<br>s=0 **OB6**, OB2 to 5, OB13 to 10<br>s=1 OB2 to 5, **OB6**, OB13 to 10<br>(OBs 2 to 6 cannot be interrupted!) |
| | N=No Y=Yes | |
| **Block Identifier: TFB:** | | **Timer-Function Block** |
| OB10 | p | Interval (msec.) during which OB10 to 13 is called and processed<br>(p=0 to 65535 (programmable in 10-msec. intervals)) |
| OB11 | p | |
| OB12 | p | |
| OB13 | p | |

1 Additionally, set switch for Default/Overall Reset on the control panel of the CPU to "RE"

| Parameter | Argument | Meaning |
|---|---|---|
| **Block Identifier: PFB:** | | **Placement of FB** |
| SFB | p q | "***S***ubstitute ***FB***"<br>Replace number p of the integral FB p (COMPR or DELETE) with the number q |
| p=238, 239 | q=0 to 239, 252 to 255 | |
| **Block Identifier: CLP:** | | **Clock Parameters (only in the case of CPU 943/944 with two interfaces)** |
| CLK | DBxDWy or MBy | "***CL***oc***K*** Data" start of clock data area |
| STW | DBxDWy or MBy | "***ST***atus ***W***ord" Position of the status word |
| SET | wd dd.mm.yy hh:mn:ss AM/PM[1] | Set clock time, date |
| TIS | wd dd.mm. hh:mn:ss AM/PM[1] | "***T***imer ***I***nterrupt ***S***et" |
| OHS | hhhhhh:mn:ss[2] | "***O***perating ***H***ours counter ***S***et" |
| OHE | J/Y/N | "***O***perating ***H***our counter ***E***nable" |
| STP | J/Y/N | "***ST***O***P***" Update clock in STOP state |
| SAV | J/Y/N | ***SAV***e Save clock time after last RUN STOP or Power OFF |
| CF | p | "***C***orrection ***F***actor" Enter correction factor |

| | | | |
|---|---|---|---|
| wd | = | 1 to 7 (Weekday=Su to Sa) | p=– 400 to 400 |
| tt | = | 01 to 31 (Day) | x=2 to 255 |
| mm | = | 01 to 12 (Month) | y=0 to 255 |
| yy | = | 0 to 99 (Year) | y/Y=yes |
| hh | = | 1 to 12 (AM/PM) 00 to 23 | n/N=no |
| mn | = | 00 to 59 (Minutes) | |
| ss | = | 00 to 59 (Seconds) | |
| hhhhhh | = | 000000 to 999999 (Hours) | |

| Block identifier: ERT | | Error Return |
|---|---|---|
| ERR | MBx or DByDWZ | "***ERR***ORs" Position of the error code<br>(x=0 to 236<br>y=2 to 255<br>z=0 to 255) |

1 If an argument (e.g. Weekday) is not to be transferred, enter XX! The clock will then continue with the current value. If you specify AM or PM after the clock time, the clock will operate in the relevant 12-hour mode. If you omit this argument, the clock will operate in 24-hr mode.

2 If an argument (e.g. minute) is not to be transferred, enter XX! The clock will then continue with the current value.

There is no parameter block for defining interprocessor communication flags. If you require these flags for the use of particular CPs, proceed as described in Section 12.1.1. Only after defining the interprocessor communication flags can you start assigning parameters to the functions described in this section.

### 11.3.8 DB1 Programming Example

The following example of a DB1 program shows you the complete DB1 parameterization once again.
The following have been parameterized:

- System characteristics
- Data interchange over SINEC L1
- Time-driven processing
- Positioning of FBs
- Integral clock
- Address for parameter error code

| STL | Explanation |
|---|---|
| 0: KC ='DB1 | DB1 header |
| 12: KC ='# Sys. characteristics # | Comment |
| 24: KC ='SDP: WD_scan monit. 500 | Header and parameter cycle monitoring |
| 36: KC ='RDLY_run delay 1000 | Restart delay |
| 48: KC ='RT_resident_timers n | Retentive feature of the timers (half or all) |
| 60: KC ='RC_resident_counters n | Retentive feature of the counters ( " ) |
| 72: KC ='RF_resident_flags n | Retentive feature of the flags ( " ) |
| 84: KC ='PROT_software protection n | Software protection active or not |
| 96: KC ='PIO_inhibit n | Disable process image of the outputs |
| 108: KC ='PII_inhibit n | Disable process image of the inputs |
| 120: KC ='PRIO_OB6_priority 0 | OB 6 priority over OB 2 |
| 132: KC =''; | Block end identifier |
| 144: KC ='# Sinec L1 parameters # | Comment |
| 156: KC ='SL1: SLN_slave number 2 | Sinec L1 parameter; slave number |
| 168: KC ='PGN_PG_bus number 2 | Programmer bus number |
| 180: KC ='SM_send mlbx. DB60DW40 | Position of the send mailbox |
| 192: KC ='RM_rec. mlbx. DB60DW0 | Position of the receive mailbox |
| 204: KC ='CBS_CB_send MB61 | Coordination byte send |
| 216: KC ='CBR_CB_rec. MB60 | Coordination byte receive |
| 228: KC =''; | Block end identifier |
| 240: KC ='# Time-driven proc. # | Comment |
| 252: KC ='TFB: | Block identifier for time-controlled processing |
| 264: KC ='OB10_interval 400 | Call interval OB 10 |
| 276: KC ='OB11_interval 300 | Call interval OB 11 |
| 288: KC ='OB12_interval 200 | Call interval OB 12 |
| 300: KC ='OB13_interval 100 | Call interval OB 13 |
| 312: KC =''; | Block end identifier |
| 324: KC ='# Placement of FB # | Comment |
| 336: KC ='PFB: | Block identifier placement of FB 238/239 |
| 348: KC ='SFB_Compr 238 210 | FB 238 receives number 210 |
| 360: KC ='SFB_Delete 239 211 | FB 239 receives number 211 |
| 372: KC =''; | Block end identifier |
| 384: KC ='# HW clock params. # | Comment |
| 396: KC ='CLP: | Block identifier HW CLOCK |
| 408: KC ='CLK_clock data DB2DW0 | Start of the clock data area |
| 420: KC ='STW_status word MW190 | Status word for clock |

## 12 Communications Capabilities


12.1 Data Interchange . . . . . 12- 1
12.1.1 Interprocessor Communication Flags . . . . . 12- 1
12.1.2 Page Frame Addressing . . . . . 12- 7

12.2 SINEC L1 Local Area Network . . . . . 12- 7
12.2.1 Principle of Operation of the SINEC L1 Local Area Network . . . . . 12- 8
12.2.2 Coordinating Data Interchange in the Control Program . . . . . 12- 9
12.2.3 Assigning Parameters to the S5-115U for Data Interchange . . . . . 12- 12

12.3 Point-to-Point Connection . . . . . 12- 16
12.3.1 Connecting a Communications Partner . . . . . 12- 17
12.3.2 Setting Parameters and Operation . . . . . 12- 18

12.4 ASCII Driver (for CPU 943/944 with Two Serial Interfaces Only) . . . 12- 20
12.4.1 Data Traffic . . . . . 12- 21
12.4.2 Coordination Bytes . . . . . 12- 23
12.4.3 Mode . . . . . 12- 24
12.4.4 ASCII Parameter Set . . . . . 12- 26
12.4.5 Assigning Parameters . . . . . 12- 29
12.4.6 Sample Program for ASCII Driver . . . . . 12- 30

12.5 Communications Link Using the 3964/3964R Communications Protocol (for CPU 944 with Two Serial Ports Only) . . . . . 12- 38
12.5.1 Data Interchange over the SI 2 Interface . . . . . 12- 40
12.5.2 Assigning a Mode Number (System Data Word 55, $EA6E_H$) . . . . . 12- 41
12.5.3 Assigning the Driver Number for a Communications Link . . . . . 12- 42
12.5.4 Transmission . . . . . 12- 42
12.5.5 Sample Program for Transmitting Data . . . . . 12- 53

## Figures


12-1. Interprocessor Communication Flag Areas when Several CPs Are Used . . 12- 6
12-2. PLCs Linked over the SINEC L1 Local Area Network . . . . . 12- 8
12-3. Data Transport . . . . . 12- 9
12-4. Structure of the Send and Receive Mailboxes . . . . . 12- 10
12-5. Structure of the Coordination Bytes for "Send" and "Receive" . . . . . 12- 11
12-6. Pin Assignments when Using Direct Line . . . . . 12- 17
12-7. Structure of the "Receive" Coordination Byte . . . . . 12- 18
12-8. Structure of the "Send" Coordination Byte . . . . . 12- 19
12-9. Example of Connector Pin Assignments for Printer Cable . . . . . 12- 21
12-10. Example of Data Transport . . . . . 12- 21
12-11. Coordination Byte Format . . . . . 12- 23
12-12. Connector Pin Assignments on the Connecting Cable between the CPU 944/SI 2 and the PT88 or PT88 S-21 Printer (TTY). . . . . 12- 30
12-13. ASCII Driver Program Structure for RESTART . . . . . 12- 31
12-14. Structure of the Cyclic ASCII Driver Program . . . . . 12- 32
12-15. Communications Link between a CPU 944 with Two Serial Ports and a Remote Node . . . . . 12- 38
12-16. Communications Link between a CPU 944 and a CP 525 . . . . . 12- 39
12-17. Communications Link between a CPU 944 and a CP 523 . . . . . 12- 39
12-18. Data Interchange over the SI 2 Interface . . . . . 12- 40
12-19. Structure of the Send Mailbox . . . . . 12- 49
12-20. Structure of the CBS . . . . . 12- 49
12-21. Structure of the CBR . . . . . 12- 51


## Tables


12-1. Definition of Interprocessor Communication Flags When Two CPs Are Used . . . . . 12- 6
12-2. Destination and Source Number Assignment . . . . . 12- 10
12-3. SINEC L1 Parameter Block . . . . . 12- 12
12-4. Assigning Parameters as a Flag Byte . . . . . 12- 13
12-5. Assigning Parameters as a Data Byte . . . . . 12- 13
12-6. Communications Partners (Slaves) for Point-to-Point Connection . . . . . 12- 16
12-7. Description of System Data Word 46 . . . . . 12- 20
12-8. Error Information in the Coordination Bytes . . . . . 12- 24
12-9. Description of the Mode Numbers . . . . . 12- 25
12-10. ASCII Parameter Set . . . . . 12- 26
12-11. Character Frame and Order of Bits on the Line in the Case of ASCII Transmission (Depending on Word 2 of the ASCII Parameter Set) . . . . . 12- 28
12-12. Parameter Block for the ASCII Driver . . . . . 12- 29
12-13. Parameter Block Contents . . . . . 12- 29
12-14. Parameter Block for a Communications Link . . . . . 12- 41
12-15. Meaning of the Mode Number . . . . . 12- 41
12-16. System Data Word SD 46 . . . . . 12- 42
12-17. Parameter Set . . . . . 12- 47
12-18. Character Frame and Order of Bits on the Line in the Case of a Computer Link (Depending on Word 2 of the ASCII Parameter Set) . . . . . 12- 48
12-19. Error Codes in the "Coordination Byte for Send" . . . . . 12- 50
12-20. Error Codes in the "Coordination Byte for Receive" . . . . . 12- 52

# 12 Communications Capabilities

The processors of individual modules (CPUs, CPs, or intelligent I/Os) can exchange information in different ways.

## 12.1 Data Interchange

There are three ways of organizing data interchange between the S5-115U CPUs and CPs/IPs:
- Data interchange over interprocessor communication flags (e.g. in the case of CP 525 and CP 526)
- Data interchange over dual-port RAM (page addressing)
- Data interchange over the I/O area (e.g. CP 523; PROFIBUS with global and cyclic I/O)

Interprocessor communication flags and page addressing are described in the following subsections; see the "SINEC L2 Local Area Network" Manual for more detailed information on data interchange over the I/O area using PROFIBUS.

### 12.1.1 Interprocessor Communication Flags

Binary signals are exchanged between central processing units (CPU 941 to CPU 944) and some communications processors, e.g., CP 526, via interprocessor communication flags. The CPU processes interprocessor communication flags like normal flags. However, they are stored in a special 256-byte memory area between the addresses $F200_H$ and $F2FF_H$.
The control program must identify interprocessor communication flags byte by byte in data block DB1 as input flags or output flags.

The transfer of interprocessor communication flags is similar to the transfer of inputs and outputs to and from the process images. The procedure is as follows:
- The interprocessor communication input flags are read in and stored in the appropriate memory area prior to program scanning.
- Interprocessor communication output flags are transferred to the appropriate CPs at the end of program scanning.

Interprocessor communication output flags can be treated like normal flags.
Interprocessor communication input flags should be scanned only, since the setting or resetting of bits can be canceled during the next data transfer.

#### Definition of the Interprocessor Communication Flags in DB1

You can program DB1 in the following two ways:
- with the help of a screen form on a programmer
- through direct input of data words

**Note**

If you are using interprocessor communication flags and you use DB1 as parameter DB for internal functions (see Chapter 11), then proceed as follows:

Overall reset
Transfer integrated DB1 to the programmer
Insert interprocessor communication flag agreements (as described below) ***before*** the DB1 parameters awaiting interpretation (see Chapter 11)
Modify and expand the other DB1 parameters (see Chapter 11)
Transfer the modified and expanded DB1 parameters to the PLC

The first three data words form the header ID. Always program them as follows:

| | | | |
|---|---|---|---|
| DW 0 : | KH=4D41 | or | KS=MA |
| DW 1 : | KH=534B | or | KS=SK |
| DW 2 : | KH=3031 | or | KS=01 |

After specifying an ID for the operand area, enter the numbers of all flag bytes used. Conclude the interprocessor communication flag list with an end ID. The IDs are as follows:

| | | | | |
|---|---|---|---|---|
| KH | = | CE00 | for | interprocessor communication input flags |
| KH | = | CA00 | for | interprocessor communication output flags |
| KH | = | EEEE | for | end |

You can use a total of 256 bytes as interprocessor communication flags. Number the bytes in relation to the start address of the interprocessor communication flag area (FB0 to FB255). The end identifier can be followed by the DB1 section in which internal functions are initialized (see Chapter 11).

**Example:**

Define flags bytes FB10, FB20, and FB30 as interprocessor communication input flags. Define flag bytes FB11 and FB22 as interprocessor communication output flags.

Assign DB1 as follows:

| | | | | | |
|---|---|---|---|---|---|
| DW | 0 | : | KH = | 4D41 | |
| | 1 | : | KH = | 534B | Header ID |
| | 2 | : | KH = | 3031 | (SCREEN 01) |
| DW | 3 | : | KH = | CE00 | |
| | 4 | : | KF = | +10 | Interprocessor communication input |
| | 5 | : | KF = | +20 | input flags |
| | 6 | : | KF = | +30 | |
| DW | 7 | : | KH = | CA00 | |
| | 8 | : | KF = | +11 | Interprocessor communication |
| | 9 | : | KF = | +22 | output flags |
| DW | 10 | : | KH = | EEEE | End ID |

The following points apply to the assignment of DB1:

- The parameter data to be interpreted must always be preceded by interprocessor communication flag definitions.
- You can enter interprocessor communication flag areas in any order.
- You can enter the byte numbers for an area in any order.
- The CPU accepts the entries in DB1 only during Manual Restart. You must therefore execute a program restart each time you modify DB1.

**Signal Exchange with a CP**

Set jumpers on the CP to enable the area required as interprocessor communication flag bytes. The jumpers divide the area between bytes 0 and 255 into eight blocks of 32 bytes each.
Normally the entire interprocessor communication flag area is enabled. Setting is necessary only when you use several CPs with interprocessor communication flags.
Specify the desired interprocessor communication flags in DB1. The bytes must be in the set area. You can choose any bytes from this area. However, use only as many bytes as necessary to keep the transfer time as short as possible.

**Example:**
20 interprocessor communication flag bytes are needed for a signal exchange:

- 14 bytes to transfer information to the CP
- 6 bytes to fetch information from the CP

The jumper setting on the CP enables the area between byte 128 and byte 159.
The interprocessor communication flags are defined in DB1 as follows:
Outputs: FY 128 to 141
Inputs: FY 142 to 147

The words in the DB are assigned as follows:

```
DW  0   :   KH  =   4D41
    1   :   KH  =   534B        Header ID
    2   :   KH  =   3031

DW  3   :   KH  =   CE00
    4   :   KF  =   +142
    5   :   KF  =   +143
    6   :   KF  =   +144        Interprocessor communication input
    .                           flags
    .
    .
DW  9   :   KF  =   +147

DW 10   :   KH  =   CA00
   11   :   KF  =   +128
   12   :   KF  =   +129        Interprocessor communication output
    .       .       .          flags
    .       .       .
    .       .       .
DW 24       KF  =   +141

DW 25   :   KH  =   EEEE        End ID
```

**Special Points to Observe when Using the CP 525 and CP 526 in RESTART Mode**

## Note

If the CP 525 and CP 526 are used in the S5-115U, the interprocessor communication flag area enabled on the CPs in RESTART mode should be reset on restart in connection with the following CP functions:

CP 525 (6ES5 525-3UA11):

- Component: Event printer if group disable bits are used
- Component: Operator-process communication and visualization with the 3975 display unit if bit set and reset commands are used

  general: Group disable bits should always be located in the interprocessor communication flag area enabled per jumper setting.

CP 526 (6ES5 526-3Lxxx):

- Basic board: If bit set and reset commands are used

Before synchronizing the CPs, an FB should be called in OB21/22. This FB should be programmed as shown in the following example:

**Example:**

Function block FBxxx (e.g. FB11) for resetting the interprocessor communication flag area on a CP.
The communication flag areas enabled by jumpers on the CP can be reset with the following block. This FB must be specified with its starting flag byte (V-FY) and end flag byte (B - FY) for each contiguous communication flag area.
If a flag byte that does not define an area boundary is specified here, the entire area is still reset.

| | | | |
|---|---|---|---|
| V-FY | : | FY35 | (from) |
| B-FY | : | FY165 | (to) |

This resets the communication flag area from flag byte FY 32 to flag FY 191. This area must naturally have been enabled on the CP.

| FB11 | STL | | Description |
|---|---|---|---|
| NAME | : K-FY | LOE | FB FOR RESETTING IPC FLAGS |
| DECL | : V-FY | I/Q/D/B/T/C: I | BI/BY/W/D: BY |
| DECL | : B-FY | I/Q/D/B/T/C: I | BI/BY/W/D: BY |
| | : | LW =V-MB | COMPUTE STARTING ADDRESS |
| | : | L KH00FF | |
| | : | AW | |
| | : | L KHF200 | |
| | : | OW | |
| | : | L KHFFE0 | |
| | : | AW | |
| | : | T FW250 | COMPUTE STARTING ADDRESS |
| | : | LW =B-MB | COMPUTE END ADDRESS |
| | : | L KH00FF | |
| | : | AW | |
| | : | L KHF200 | |
| | : | OW | |
| | : | L KH001F | |
| | : | OW | |
| | : | L KHFFFE | |
| | : | AW | |
| | : | T FW252 | |
| M001 | : | | END ADDRESS |
| | : | L KH0000 | |
| | : | L FW250 | LOOP FOR |
| | : | TIR 2 | RESETTING IPC |
| | : | L FW252 | FLAGS |
| | : | !=F | |
| | : | BEC | |
| | : | L FW250 | |
| | : | ADD KF+2 | |
| | : | T FW250 | |
| | : | JU =M001 | |
| | : | BE | |

**Note**

Make sure that "SYSTEM OPERATIONS" are enabled in the Defaults form.

**Signal Exchange with Several CPs**

If one CPU addresses several CPs, one or more interprocessor communication flag areas must be enabled on each CP. When setting the jumpers on the CPs, please note the following points:

- The areas on the individual CPs must not overlap (to prevent duplicate address assignment).
- The areas on the individual CPs do not have to be assigned consecutively (see Figure 12-1).

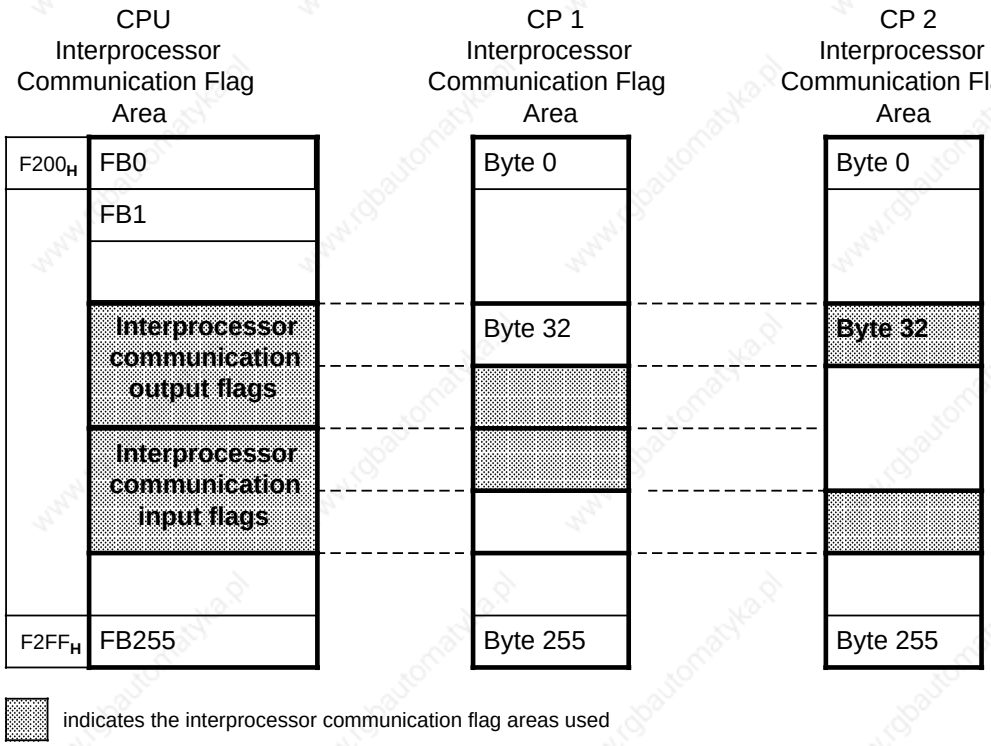



**Figure 12-1. Interprocessor Communication Flag Areas when Several CPs Are Used**

Define the interprocessor communication flag bytes in data block DB1 as described above.

**Example:**

Use one CPU to address two CPs. Table12-1 shows the flag bytes needed and possible numbering.

**Table 12-1. Definition of Interprocessor Communication Flags when Two CPs Are Used**

| CPs | Number of Control Bytes (Outputs) | Number of Scan Bytes (Inputs) | CP Flag Areas Set | CPU Interprocessor Communication Output Flags | CPU Interprocessor Communication Input Flags |
|---|---|---|---|---|---|
| CP 1 | 8 | 4 | Bytes 128 to 159 | FB 128 to 135 | FB 156 to 159 |
| CP 2 | 6 | 10 | Bytes 160 to 191 | FB 170 to 175 | FB 160 to 169 |

### 12.1.2 Page Frame Addressing

Modules that can be programmed and modules to which parameters can be assigned (CPs and IPs) process complex jobs in the SIMATIC S5 system. These modules have a one-kilobyte dual port RAM for data exchange with the PLC. In the CPU, this interface memory is assigned an address area that can be addressed linearly or via a page frame.[1]
For linear addressing, each interface needs a one-kilobyte area in user memory. In order to prevent a loss of capacity when several CPs are used, all CPs and some intelligent I/Os are addressed via a page frame on the S5-115U. In addition to the memory area $F400_H$ to $F7FF_H$ for the page frame, only one memory location is needed in the internal register to specify an interface number (address $FEFF_H$) between 0 and 255.

The same numbers are set on the module. This procedure determines which interface is addressed through the page frame. If a module has two interfaces, they are numbered in ascending order.

Data handling blocks are used for data exchange (see Subsection 11.1.3). They must be called by the control program. The essential information for a particular job is entered in the parameter list of the handling block.

## 12.2 SINEC L1 Local Area Network

SINEC L1 is a communications system that networks SIMATIC S5 programmable controllers of the U range. It works according to the master-slave principle:

- The **master** is a separate PLC that handles the entire coordination and monitoring of data traffic in the local area network.
  The master PLC must have a CP 530 communications processor.
- A **slave** can be any PLC.

"Data handling blocks" which support communications with the CP 530 are integrated in the central processing units of the S5-115U (see Section 11.1.3).

1 A page frame is a specific area of the user memory.

### 12.2.1 Principle of Operation of the SINEC L1 Local Area Network

You can connect one master and up to 30 slaves to the SINEC L1 local area network. Each node, master or slave, needs a BT 777 transceiver as level converter.
The BT 777 can be connected to:

- the slaves' programmer port (in which case, data is interchanged via Send/Receive mailboxes as described in the following)
  or
- the (master or slave) CP 530's SINEC L1 interface (in which case, you must proceed as described in the "SINEC L1 Local Area Network" manual (6ES5 998-7LA21). In this case, the data is interchanged via data handling blocks).

Data is transferred over a 4-wire shielded cable that interconnects the various transceivers.

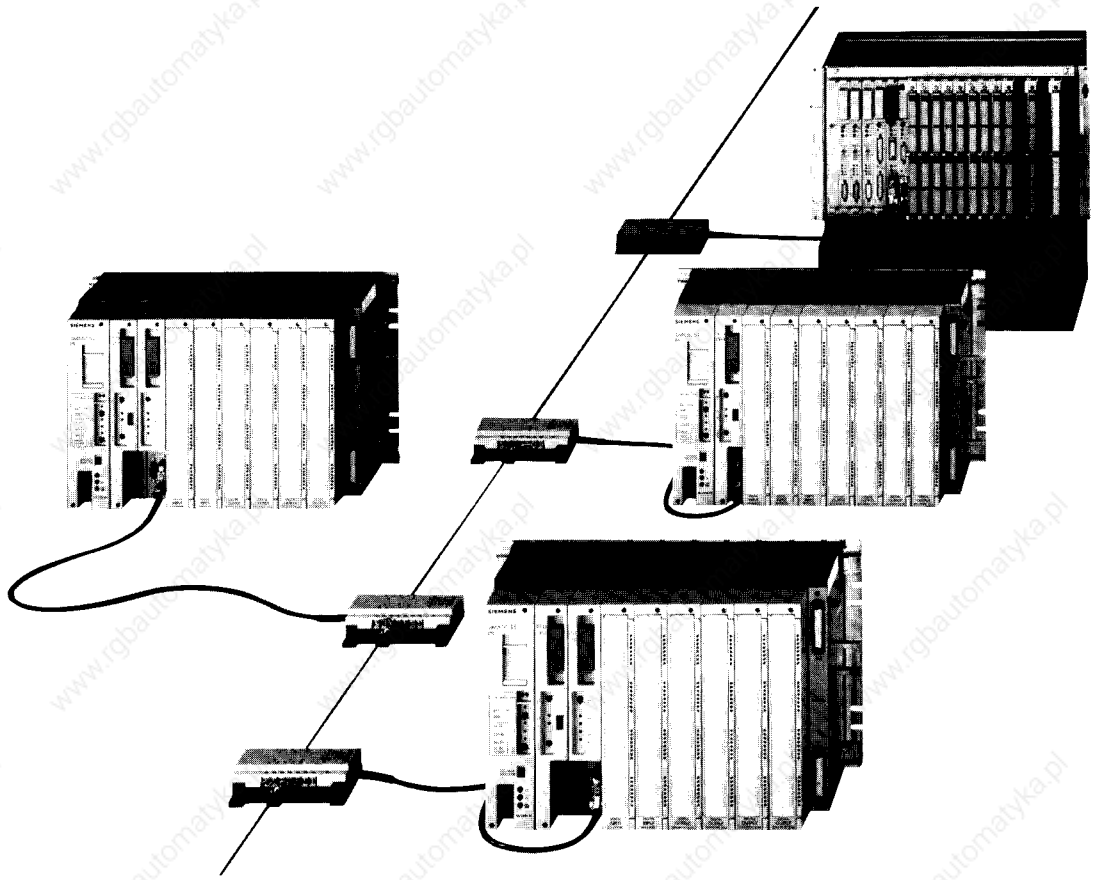


**Figure 12-2. PLCs Linked over the SINEC L1 Local Area Network**

You can transfer data over the SINEC L1 local area network in the following two ways:
- from one node to another
  - master slave
  - slave master
  - slave slave
- from one node to all other nodes simultaneously (broadcast).

The following data can be transmitted:
- signal states of inputs, outputs, and flags;
- contents of data words.

Besides data, you can also transmit programmer functions on the SINEC L1 local area network. A programmer that is connected to the master's CP 530 can address individual slaves (see the SINEC L1 manual 6ES5 998-7LA21).

### 12.2.2 Coordinating Data Interchange in the Control Program

A slave needs the following to interchange data:
- a slave number (1 to 30)
- a Send mailbox (SF)
- a Receive mailbox (EF)
- coordination bytes

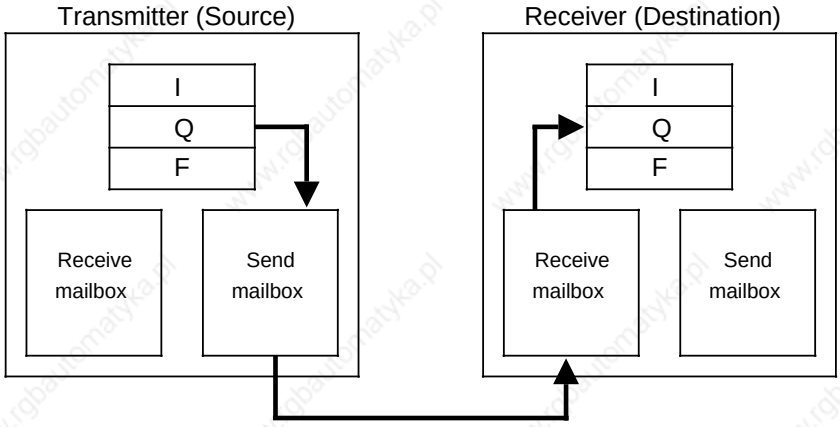



**Figure 12-3. Data Transport**

**Send and Receive Mailboxes**

The Send and Receive mailboxes contain send and receive data. They can hold up to 64 bytes of information. They also contain the following:

- Length of the data packet (1 to 64 bytes)
- Type of mailbox
  - The Send mailbox specifies the destination number.
  - The Receive mailbox contains the source number.

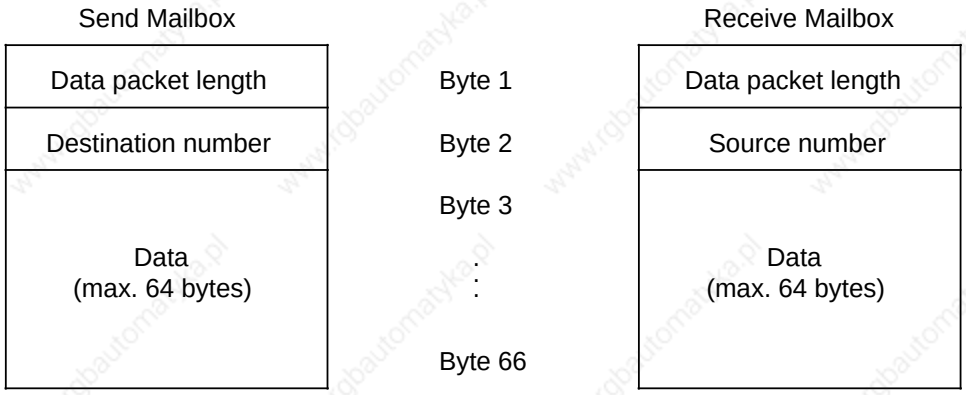



**Figure 12-4. Structure of the Send and Receive Mailboxes**

The source or destination number indicates the "device" with which you want to communicate. Refer to Table 12-2 for the meaning of these numbers.

**Table 12-2. Destination and Source Number Assignment**

| Assignment | Partner |
|---|---|
| 0 | Master |
| 1 to 30 | Slave |
| 31 | Broadcast |

Use the control program to access the mailboxes.

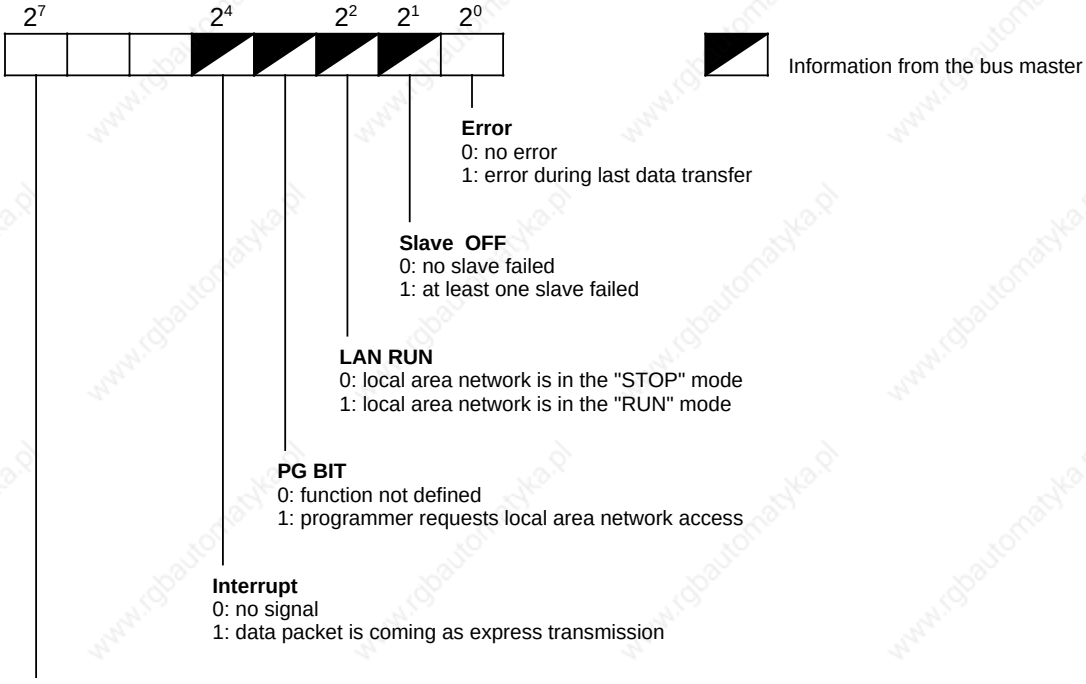
You can assign to the location of the mailboxes.
You can define the starting addresses of the mailboxes in either of the following ways:

- Specify a data block and a data word.
- Specify a flag word.

**Coordination Bytes**

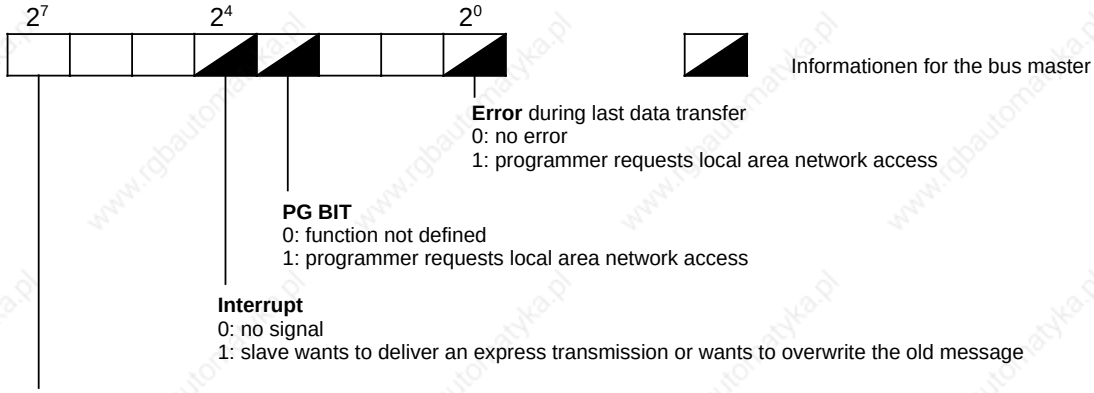
Coordination bytes form the interface to the PLC's operating system.
The control programs for the slaves use these bytes to track the flow of local area network traffic and to influence it.
The following Figures describe the meanings of individual bits.

Coordination Byte for "Receive" (CBR) (Flag byte or high byte in data word)




**REC-PERM**
0: The program can fetch data from the Receive mailbox. The operating system has no access.
1: The operating system can retrieve data from the local area network via the Receive mailbox. The program has no access.
If REC-PERM="1," the operating system fills the Receive mailbox with data. Then the operating system resets REC-PERM to "0" (=0).

Coordination Byte for "Send" (CBS) (Flag byte or high byte in data word)




**SEND-PERM**
0: The program can process the Send mailbox. The operating system has no access.
1: The Send mailbox is enabled to transmit on the local area network. The programm has no access.
SEND-PERM="1" causes the operating system to transmit the contents of the Send mailbox. Afterwards the operating system resets the SEND-PERM bit to "0".

**Figure 12-5. Structure of the Coordination Bytes for "Send" and "Receive"**

### 12.2.3 Assigning Parameters to the S5-115U for Data Interchange

You must always specify the following in the program:
- local slave number
- the data or flag areas assigned for the Send **and** Receive mailboxes
- the location of the coordination bytes

You can also define the following in the program (if required):
- your own programmer No. for programmer bus functions

On the S5-115U, you can specify the location of the coordination bytes, the Send mailbox and the Receive mailbox.
Programming is done either
- in DB1 (see Section 11.3) or
- in a function block that is called by one of the two restart organization blocks (OB21 or OB22). Use the "TNB" or "TBS" block transfer operation to store the appropriate parameters in the system data area. The SINEC L1 parameter block begins at system data word RS57 (see Table 12-3).

**Table 12-3. SINEC L1 Parameter Block**

| System Data Word | High-Order Byte | Low-Order Byte | Absolute Addresses |
|---|---|---|---|
| SD 57 | PG number * (1 to 30) | Slave number (1 to 30) | EA72 EA73 |
| SD 58 | CBR data ID ** | CBR DB or flag byte | EA74 EA75 |
| SD 59 | CBR data word | CBS data ID ** | EA76 EA77 |
| SD 60 | CBS DB or flag byte | CBS data word | EA78 EA79 |
| SD 61 | SMB data ID ** | SMB DB or flag byte | EA7A EA7B |
| SD 62 | SMB data word | RMB data ID ** | EA7C EA7D |
| SD 63 | RMB DB or flag byte | RMB data word | EA7E EA7F |

***** You need a PG number if you want to transmit programmer functions over the SINEC L1 local area network.
Note carefully: Slave number "0" in the low-order byte is indicative of a master function. In this case, no PG/OP functions may be forwarded over the CPU 943's or CPU 944's SI 2 interface (see Section 12.3.2)!
The PG number is retained in the event of a CPU Overall Reset via the programmer bus.

****** Flag or data byte; see Tables 12-4 and 12-5

Specify the location of the coordination bytes and the start addresses of the Send and Receive mailboxes with three bytes each.
You can specify this information in the function block.
Define each byte either as a flag byte or as the high byte of a data word.

**Table 12-4. Assigning Parameters as a Flag Byte**

| Meaning | Parameter | Address * | | | |
|---|---|---|---|---|---|
| | | CBR | CBS | SMB | RMB |
| Data ID "flag" | $4D_H$ (M * *) | EA74 | EA77 | EA7A | EA7D |
| Flag byte no. | 0 to 255 | EA75 | EA78 | EA7B | EA7E |
| irrelevant | | EA76 | EA79 | EA7C | EA7F |

**Table 12-5. Assigning Parameters as a Data Byte**

| Meaning | Parameter | Address * | | | |
|---|---|---|---|---|---|
| | | CBR | CBS | SMB | RMB |
| Data ID "data" | $44_H$ (D * *) | EA74 | EA77 | EA7A | EA7D |
| Data block no. | 2 to 255 | EA75 | EA78 | EA7B | EA7E |
| Data word no. | 0 to 255 | EA76 | EA79 | EA7C | EA7F |

***** Destination addresses in the system data area

*** *** ASCII coded data ID

### Overflow

If data packets are received that are longer than the Receive mailbox, writing does ***not*** extend beyond the end of the Receive mailbox. There is no overflow signal.
The end of the receive area is flag byte 255 in the flag area or the last data word (in the data block).

**Example of SINEC L1 Parameter Assignments:**

Set parameters in OB22 (OB21). FB 255 is used to handle parameter entry.
The formal operands indicate the type and number of the coordination bytes (CBR, CBS) and of the data mailboxes (RMB, SMB), e.g., TCBR is a "receive" coordination byte.

| OB21/OB22 STL | Description |
|---|---|
| : JU FB 255<br>NAME : L1 PARAM<br>SLNO : KY 0,1<br>TCBR : KS FY<br>NCBR : KY 100,0<br>TCBS : KS FY<br>NCBS : KY 101,0<br>TSMB : KS DB<br>NSMB : KY 2,1<br>TRMB : KS DB<br>NRMB : KY 2,40<br>: BE | <br><br>SLAVE 1<br>CBR : FLAG AREA<br>FY 100<br>CBS : FLAG AREA<br>FY 101<br>SMB : DATA BLOCK<br>DB2 BEG. DW1<br>RMB : DATA BLOCK<br>DB2 BEG. DW40 |

Parameter description:

SLNO: PG bus address/data slave address (KY a, b):
a) PG bus address
b) Data slave number

TCBR/NCBR: Type of COOR byte for RECEIVE/SEND (KS):
Options are FY =^ Flag byte
DW =^ Data word (left)

TCBS/NCBS: Number or address of COOR byte for RECEIVE/SEND (KY a, b):
a) For type FY =^ Number of the flag byte
For type DW =^ Number of the data block
b) For type FY =^ "0"
For type DW =^ Number of the data word (left)

TSMB/NSMB: Type of SEND/RECEIVE MAILBOX (KS):
Options are FY =^ Flag byte
DW =^ Data word (left)

TRMB/NRMB: Number of the SEND/REVCEIVE MAILBOX (KY a, b):
a) Type FY =^ Number of the flag byte with which the Send/Receive mailbox begins
Type DB =^ Number of the data block
b) Type FY =^ "0"
Type DB =^ Number of the data word with which the Send/Receive mailbox begins

| FB255 STL | Description |
|---|---|
| NAME :L1 PARAM | |
| DECL :SLNO I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KS/KG: KY | |
| DECL :TCBR I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KS/KG: KS | |
| DECL :NCBR I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KS/KG: KY | |
| DECL :TCBS I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KS/KG: KS | |
| DECL :NCBS I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KS/KG: KY | |
| DECL :TSMB I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KS/KG: KS | |
| DECL :NSMB I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KS/KG: KY | |
| DECL :TRMB I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KS/KG: KS | |
| DECL :NRMB I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KS/KG: KY | |
| :LW =SLNO | L1-PG-BUS/L1-DATA BUS SLAVE NO. |
| :T FW200 | |
| : | |
| : | |
| :LW =TCBR | TYPE OF COORDINATION BYTE "R" |
| :T FW202 | (RECEIVE) |
| : | |
| :LW =NCBR | ADDRESS OF THE CBR |
| :T FW203 | DB OR FY NO./DW NO. |
| : | |
| :LW =TCBS | TYPE OF COORDINATION BYTE "S" |
| :T FW205 | (SEND) |
| : | |
| :LW =NCBS | ADDRESS OF THE CBS |
| :T FW206 | DB OR FY NO./DW NO. |
| : | |
| :LW =TSMB | TYPE OF SEND MAILBOX |
| :T FW208 | |
| :LW =NSMB | ADDRESS OF SEND MAILBOX |
| :T FW209 | |
| :LW =TRMB | TYPE OF RECEIVE MAILBOX |
| :T FW211 | |
| :LW =NRMB | ADDRESS OF RECEIVE MAILBOX |
| :T FW212 | |
| : | |
| :L KHEED5 | TRANSFER FROM FLAG AREA TO |
| :L KHEA7F | SYSTEM DATA AREA |
| :TNB 14 | |
| : | |
| :L KH0000 | RESET SCRATCH FLAG WORDS |
| :T FW200 | |
| :T FW202 | |
| :T FW204 | |
| :T FW206 | |
| :T FW208 | |
| :T FW210 | |
| :T FW212 | |
| : | |
| :BE | |

Use the following example to initialize an S5-115U as PG bus node:

**Example:** Initializing an S5-115U CPU connected only as programmer bus node to the SINEC L1 LAN.
The function block for programmer address assignments (FB1) is invoked in the restart OBs (OB21 and OB22).

| OB21/OB22 STL | Description |
|---|---|
| :<br>:JU FB 1<br>NAME : PG-ADR<br>PGAD : KY 1,0<br>:<br>:BE | CALL FB 1 FOR PG BUS ADDRESS ASSIGNMENTS<br><br>PG BUS ADDRESS OF THE CPU IS 1<br>(PERM. VALUES: 1 to 30) |

| FB1 STL | Description |
|---|---|
| NAME : PG-ADR<br>DECL : PGAD<br>: L RS 57<br>: L KH 00FF<br>: UW<br>: LW =PGAD<br>: 0W<br>:<br>: T BS 57<br>:BE | I/Q/D/B/T/C : D KM/KH/KY/KS/KF/KT/KC/KG : KY<br>LOAD SYSTEM DATA WORD 57<br>Delete old PG No.<br><br>LOAD PG BUS ADDRESS<br>SD 57 AND PG BUS ADDRESS ARE ORED,<br>THE LOW-ORDER BYTE OF RS 57 REMAINS UNCHANGED<br>WRITE RESULT OF OR OPERATION BACK TO RS 57 |

## 12.3 Point-to-Point Connection

You can connect CPU 943 (with two interfaces) and CPU 944 (with two interfaces) to a SINEC L1 slave without using an additional module. With these connections, you can transmit control and back-up information and data (see Table 12-6).

**Table 12-6. Communications Partners (Slaves) for Point-to-Point Connection**

| Communications Partners | | Connection |
|---|---|---|
| **S5-100U** | with CPU 102/103 | direct using the CPU interface |
| **S5-90U/95U/101U** | | |
| **S5-115U** | with CPU 941/942/943/944 | direct using the CPU interface or CP 530 |
| **S5-135U/150U/155U** | | using CP 530 |

### 12.3.1 Connecting a Communications Partner

You can establish connections in either of the two following ways:

- via a bus with transceivers (BT 777)
- via a direct line (Direct line connection is possible only when the controllers are no further apart than 1000 m.) Use a four-wire, shielded cable with a cross-section of at least 0.14 mm² (26 AWG). SIMATIC cable 6ES5 707-1AA00 is recommended.

**Connector Pin Assignments** (see also Appendix C)

Connect a 15-pin subminiature D connector with metal housing to each end of the cable. Figure 12-6 shows the connector pin assignments.

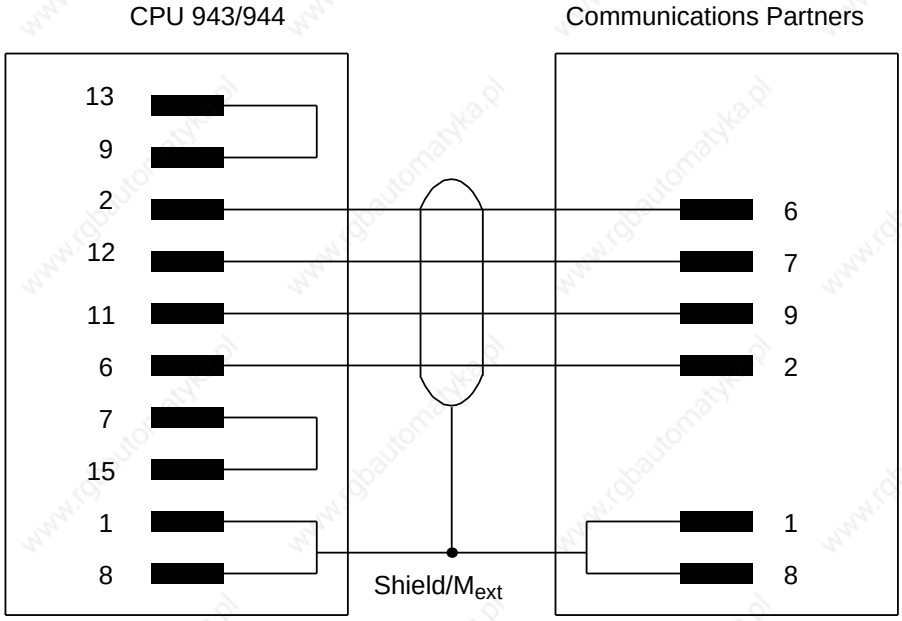



**Figure 12-6. Pin Assignments when Using Direct Line**

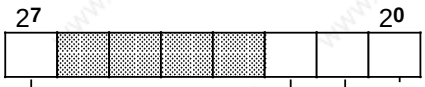
### 12.3.2 Setting Parameters and Operation

Use the SINEC L1 parameter block to initialize the interface on the CPU (see Subsection 12.2.3).
For a point-to-point connection, assign "0" as the slave number for the CPU 943/CPU 944 (master function only possible at 512). The partner in the link is always addressed as slave 1.
As long as interface SI 2 has been initialized for point-to-point connection, no programmer or OP can be operated over this connector.

> **Note**
>
> Neither broadcasting nor interrupt traffic is possible over point-to-point connections.

As with the SINEC L1 local area network, data is exchanged via Send and Receive mailboxes that the control program accesses with load and transfer operations.
The CPU operating system controls data transfer and stores the information in two coordination bytes. The control program can read and evaluate both bytes. Figures 12-7 and 12-8 explain the bits in the coordination bytes.

Receive Coordination Byte (CBR) (flag byte or high-order byte in a data word)




**Error**
0: no error
1: receive error during last data transfer

**Slave OFF**
0: normal operation
1: node failure

**LAN RUN**
0: SINEC L1 local area network is in the "STOP" mode.
1: SINEC L1 local area network is in the "RUN" mode.

**REC-PERM**
0: The program can fetch data from the Receive mailbox. The operating system has no access.
1: The operating system can fetch data from the local area network via the Receive mailbox. The program has no access.

indicates irrelevant bits

**Figure 12-7. Structure of the "Receive" Coordination Byte**

"Send" Coordination Byte (CBS) (Flag byte or high byte in data word)

**Error**
0: no error
1: send error during last data transfer

**SEND-PERM**
0: The program can process the Send mailbox. The operating system has no access.
1: The operating system sends data from the Send mailbox over the LAN. The program has no access.

indicates irrelevant bits

**Figure 12-8. Structure of the "Send" Coordination Byte**

The coordination bytes, the Send mailbox, and the Receive mailbox can be initialized in a function block (as with SINEC L1, see Section 12.2.3).
If too much data is transferred, the reaction is the same as for "Overflow" on the SINEC L1 local area network.

**Differences between Point-to-Point Connection and CP 530**

In a point-to-point connection, data is written directly into the CPU's program memory. The control program can therefore access this area only between the time the data has been received and the next frame enabled.
These steps must be coordinated over the control program.

When using a CP 530, the data in a frame is initially stored in a CP 530 buffer.
The control program triggers data transfer to the appropriate DBs. Only one read operation is needed for this transfer. While the control program is processing the DBs, the CP 530 can receive the next frame.

## 12.4 ASCII Driver (for CPU 943/944 with Two Serial Interfaces Only*)

CPUs 943/944 provide an ASCII driver for the second interface (SI 2). The ASCII driver regulates data traffic between the main processor and the second interface.

The ASCII driver functions only if you make the appropriate setting in the high-order byte of system data word 46 (see Table 12-7). Error messages are stored in the low-order byte of this system data word.

> **Note**
>
> No other functions are possible (e.g. programmer/OP) when the ASCII driver is active.

**Table 12-7. Description of System Data Word 46**

| Byte | Contents | Description |
|---|---|---|
| High-order byte | $00_H$ ** | PG/OP and SINEC L1 operation |
| | $01_H$ | ASCII driver |
| Low-order byte | | **Error Check-Back signals** |
| | $01_H$ | No ASCII driver |
| | $10_H$ | No CBS |
| | $20_H$ | No CBR |
| | $40_H$ | No CBS and no CBR |

** Default value

* with the 816-1BB11 (CPU 944) operating system submodule

**Connection**

Connector pin assignments: Printer connecting cable for CPU 943/944 (ASCII driver)/PT 88 (see also Appendix C)

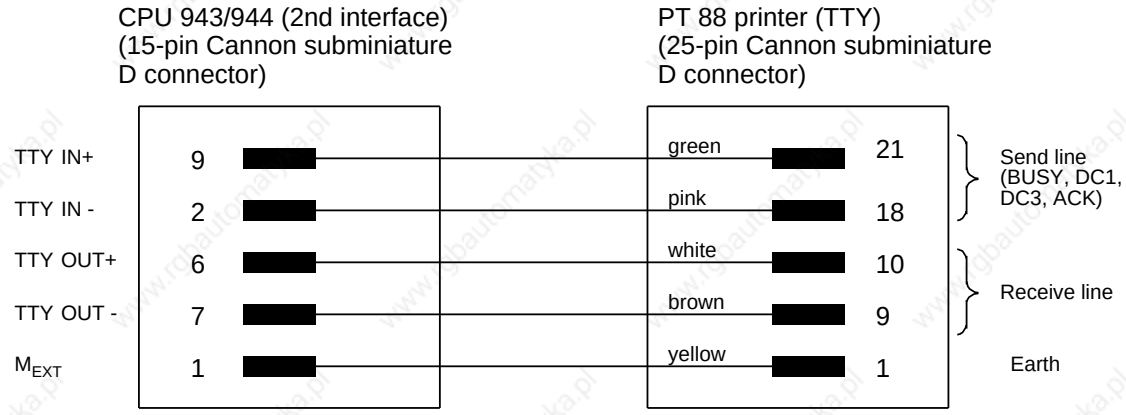



**Figure 12-9. Example of Connector Pin Assignments for Printer Cable**

> **Note**
>
> Wrong wiring can destroy the interface processor.

## 12.4.1 Data Traffic

Figure 12-10 shows how the ASCII driver functions.

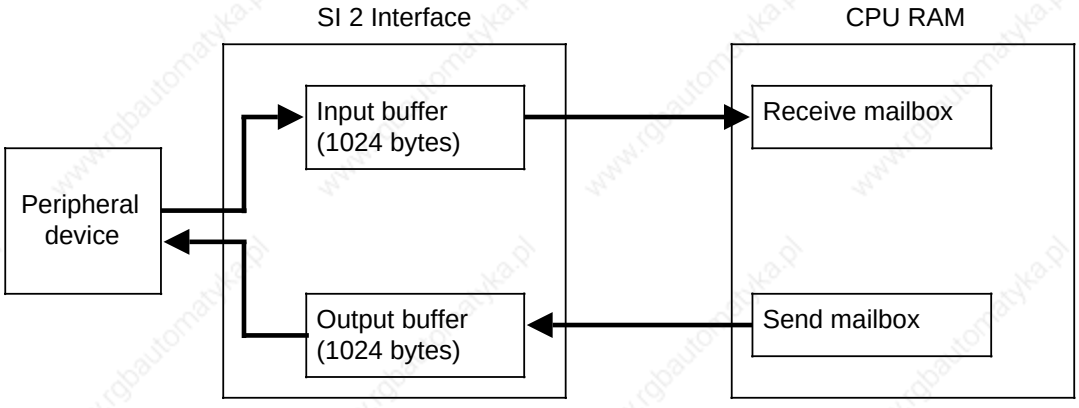



**Figure 12-10. Example of Data Transport**

Data traffic is bidirectional:

- Send
  The ASCII driver processes data in the user memory (e.g. the contents of a data block) and outputs it at the second interface.

- Receive
  An I/O device sends data in ASCII code to the second interface. The ASCII driver processes the data and stores it in internal RAM.

Data are only received every 100 ms. Please take this into account for shorter CPU cycle times.

The two memory areas in internal RAM in which the send and receive data are stored are called the Send mailbox (SMB) and the Receive mailbox (RMB).
The data can be stored either in a data block or the flag area. You must enter the relevant information in the parameter block (see Table 12-13).

Send and Receive mailboxes have the following properties:

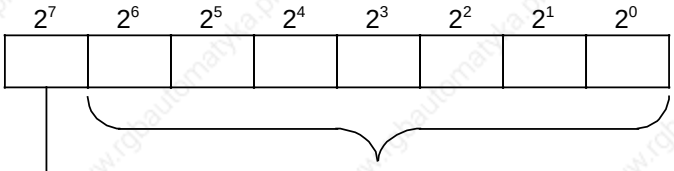
- 1024 bytes of input buffer are available in all modes.
- In modes where characters are interpreted when received (e.g. XON, XOFF), the ASCII driver can still receive data or message frames even if it has already sent XOFF to the communications partner. In this case, the ASCII driver receives data until the input buffer is full, or it receives message frames until the maximum possible number of frames has been reached.
  Example of "borderline case":
  If a received message frame is 1024 bytes long and the ASCII driver then sends XOFF, there is no way of buffering characters received from the communications parnter after sending XOFF.
- In mode 1, 7 or 8 (see Section 12.4.3), you must enter in the first word of the send mailbox the number of bytes of data to be sent.

## 12.4.2 Coordination Bytes

The ASCII driver monitors data traffic and stores status and error information in two coordination bytes, SEND (CBS) and RECEIVE (CBR).

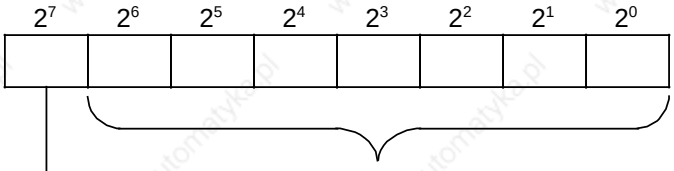
Figure 12-11 shows the format of both coordination bytes.

**SEND Coordination Byte (CBS)** (Flag byte or high-order byte in data word)




Sending permitted
Is set by the user and reset by the ASCII driver when the send procedure is finished. A leading edge at this bit activates the send procedure.

**RECEIVE Coordination Byte (CBR)** (Flag byte or high-order byte in data word)




Receive permitted
Is set by the user and reset by the ASCII driver when the Receive mailbox is full.

**Figure 12-11. Coordination Byte Format**

> **Note**
>
> As long as the "send/receive permitted" bits are set, the location of the Send and Receive mailboxes (DB or flag area) cannot be changed.

> **Note**
>
> The operating system can set or reset bits in the coordination bytes after every statement without regard to the PLC cycle. In consequence, multiple scanning of a coordination bit in a program cycle may produce different results (be careful when evaluating signal edges)!

Table 12-8 lists and explains the various error messages.

**Table 12-8. Error Information in the Coordination Bytes**

| | Contents | Description | Reaction |
|---|---|---|---|
| CBS | $07_H$ | Output buffer full | Data is rejected |
| | $0D_H$ | Parameter assignment error | |
| | $11_H$ | No Send mailbox | |
| | $13_H$ | Frame is too long | |
| CBR | $01_H$ | Character delay time exceeded | Data is valid until time is exceeded |
| | $03_H$ | Parity error | Data is rejected |
| | $05_H$ | Receive after XOFF | |
| | $07_H$ | Receive buffer full | |
| | $09_H$* | Too many frames received | Data is valid, subsequent frames are rejected |
| | $0B_H$ | Frame larger than receive mailbox | Data is rejected |
| | $0F_H$ | No Receive mailbox | |
| | $1B_H$ | Break | |

***** When CBR = 0 (e.g., receive is not possible because the PLC is in the "STOP" mode), up to 100 frames can be stored in the receive buffer.

### 12.4.3 Mode

You can use mode numbers (1 to 8) to specify the type of data traffic. DW 7 in the ASCII parameter set has a special meaning for each mode number.

A distinction is made between two types of protocol:

- noninterpreting mode (mode numbers 1, 2, and 3)
  No control characters are used in the send and receive modes.
- interpreting mode (mode numbers 4 to 8)
  An XON/XOFF protocol is used during data traffic. When the signal state of the "receive permitted" bit changes, a signal is sent to the second interface:
  - XOFF for trailing edge
  - XON for leading edge.

**Mode number**

The table below explains the mode numbers. The default refers to word 7 in the ASCII parameter set (see Table 12-10). The mode numbers must be defined in system data word 55 (see Section 12.4.5).

**Table 12-9. Description of the Mode Numbers**

| Mode | Description | Default |
|---|---|---|
| 1 | Send n bytes; n* must be specified in the first word of the Send mailbox.<br>Receive m bytes; m** is specified in the ASCII parameter set. | none<br>64 |
| 2 | Send or receive data until the end character (low-order byte) defined in the parameter set is sent or received. The end character is also transmitted. | <CR> |
| 3 | Send or receive data until the two end characters defined in the parameter set are sent or received. Both end characters are transmitted.<br>End of text is recognized only when the character defined in the high-order byte is sent or received before the character defined in the low-order byte. | <CR> <LF> |
| 4 | Same as mode 2. The following ASCII characters are interpreted upon reception:<br>RUB OUT : delete last character<br>XON : forward<br>XOFF : interrupt send and wait for XON | <CR> |
| 5 | Same as mode 3. The following ASCII characters are interpreted upon reception:<br>RUB OUT : delete last character<br>XON : forward<br>XOFF : interrupt send and wait for XON | <CR> <LF> |
| 6 | Printer Output<br>Transmit send mailbox until an end character defined in the parameter set (low byte) is reached. The end character is not transmitted.<br>Only XON and XOFF can be received. These characters are also interpreted. | <EOT> |
| 7 | Printer Output<br>Send n bytes; n must be specified in the first word of the Send mailbox. (n is not transmitted.)<br>Only XON and XOFF can be received. These characters are also interpreted. | none |
| 8 | As in mode 1; the following ASCII characters are also interpreted when received<br>RUB OUT*** : delete last character<br>XON : continue to send<br>XOFF : abort send and wait for XON | see Mode 1 |

* n is variable when sending
** m is fixed when receiving
*** If a received message frame of m bytes contains a RUB OUT, correspondingly less data is entered in the receive mailbox and the character delay time activates error 01 in CBR.

**ASCII Codes and Corresponding Hexadecimal Numbers:**

| | | | | | |
|---|---|---|---|---|---|
| RUB OUT | $7F_H$ | CR | $0D_H$ | EOT | $04_H$ |
| XON | $11_H$ | LF | $0A_H$ | ETX | $03_H$ |
| XOFF | $13_H$ | FF | $0C_H$ | | |

## 12.4.4 ASCII Parameter Set

The ASCII parameter set is used to define function parameters for the ASCII driver (see Table 12-10). There are already defaults for individual parameters according to the mode selected. The following parameters apply to the PT 88 printer.

The parameter set is read after activation of the ASCII driver or after a change of mode; data traffic at the interface must first be terminated (i.e. bit 7 of CBR=0 and bit 7 of CBS=0). The parameter set is also transferred after PLC POWER ON if the ASCII driver was previously activated.

> **Note**
>
> The default values are used only when no ASCII parameters are initialized or if these cannot be interpreted.

**Table 12-10. ASCII Parameter Set**

| Word | Description | Value Range | Default According to Mode | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 0 | Baud rate | 2 200 bauds<br>3 300 bauds<br>4 600 bauds<br>5 1200 bauds<br>6 2400 bauds<br>7 4800 bauds<br>8 9600 bauds | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| 1 | Parity | 0 even<br>1 odd<br>2 mark ("1")<br>3 space ("0")<br>4 no check | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | Data format* | 0 to 8 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 3 | Waiting time CR ** | 0 to $00FF_H$ x 10ms | x | x | x | x | x | 0 | 0 | x |
| 4 | Waiting time LF ** | 0 to $00FF_H$ x 10ms | x | x | x | x | x | 0 | 0 | x |
| 5 | Waiting time FF ** | 0 to $00FF_H$ x 10ms | x | x | x | x | x | 0 | 0 | x |
| 6 | Character delay time (receive only) | 1 to $FFFF_H$ x 10ms | 10 | 10 | 10 | 10 | 10 | x | x | 10 |

X = irrelevant
* See Table 12-11 for the meaning of data formats 0 to 8
** When sending

**Table 12-10. ASCII Parameter Set (Continued)**

| Word | Description | Value Range | Default According to Mode | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 7 | End-of-text characters/ Number of characters received | According to mode number (see Table 12-9) | | | | | | | | |
| 8 | Suppress LF | 0/1 yes/no | x | x | x | x | x | 0 | 0 | x |
| 9 | Lines per page | 1 to 255 | x | x | x | x | x | 72 | 72 | x |
| 10 | Left margin | 1 to 255 | x | x | x | x | x | 10 | 10 | x |
| 11 | Page number | o/u top/bottom | x | x | x | x | x | u | u | x |
| 12<br>.<br>.<br>. | Header/<br>Footer*** | Header 1<br>Header 2<br>Footer 1<br>Footer 2 | x | x | x | x | x | CR<br>CR<br>CR<br>CR | CR<br>CR<br>CR<br>CR | x |

X = irrelevant

*** The contents of each header and footer (max. 120 characters each) must be separated by CR.

The character delay time (word 6 of the ASCII parameter set) must conform to the following formula:

$$\text{ZVZ} \quad \frac{100}{\text{Baud rate}}$$

Example:

$$\text{Baud rate} = 4800 \frac{1}{\text{s}}$$

$$\text{ZVZ} \quad \frac{100}{4800}\ \text{s}$$

20 ms

Word 6 in the ASCII parameter set = 2

**Data Format and Character Frame**

**Table 12-11. Character Frame and Order of Bits on the Line in the Case of ASCII Transmission (Depending on Word 2 of the ASCII Parameter Set)**

| Word 2 of the ASCII Parameter Set (Data Format) | Char-acter Frame | Parity | Number of Bits per Char-acter | Order of Bits on the Line |
|---|---|---|---|---|
| 0 | 11 bits | 0 to 4* | 7 | 1 start bit, 7 data bits, 1 parity bit, 2 stop bits |
| 1 | 11 bits | 0 to 4* | 8 | 1 start bit, 8 data bits, 1 parity bit, 1 stop bit |
| 2 | 11 bits | Setting irrelevant | 8 | 1 start bit, 8 data bits, 2 stop bits |
| 3 | 10 bits | Setting irrelevant | 7 | 1 start bit, 7 data bits, 2 stop bits |
| 4 | 10 bits | 0 to 4* | 7 | 1 start bit, 7 data bits, 1 parity bit, 1 stop bit |
| 5 | 10 bits | Setting irrelevant | 8 | 1 start bit, 8 data bits, 1 stop bit |
| 6 | - | - | - | - |
| 7 As data format 0 | 11 bits | 0 to 4* | 7 | 1 start bit, 7 data bits, 1 parity bit, 2 stop bits |
| 8 As data format 1 | 11 bits | 0 to 4* | 8 | 1 start bit, 8 data bits, 1 parity bit, 1 stop bit |

* cf. Table 12-10

### 12.4.5 Assigning Parameters

You must define the position of the ASCII parameter set, the send and receive mailboxes and the coordination byte in the user program in a parameter block (see Table 12-12) located in the system data area of the CPU 943/944; you also specify the mode number there.

**Table 12-12. Parameter Block for the ASCII Driver**

| System Data Word | Absolute Address | High-Order Byte | Low-Order Byte |
|---|---|---|---|
| SD 48 | EA60 | ASCII parameter set<br>Data ID | ASCII parameter set<br>Flag byte or DB number |
| SD 49 | EA62 | ASCII parameter set<br>Data word number | SMB<br>Data ID |
| SD 50 | EA64 | SMB<br>Flag byte or DB number | SMB<br>Data word number |
| SD 51 | EA66 | RMB<br>Data ID | RMB<br>Flag byte or DB number |
| SD 52 | EA68 | RMB<br>Data word number | CBS<br>Data ID |
| SD 53 | EA6A | CBS<br>Flag byte or DB number | CBS<br>Data word number |
| SD 54 | EA6C | CBR<br>Data ID | CBR<br>Flag byte or DB number |
| SD 55 | EA6E | CBR<br>Data word number | Mode number |

Table 12-13 shows the contents of the individual bytes.

**Table 12-13. Parameter Block Contents**

| Data ID | Memory Area Specification * | |
|---|---|---|
| $4D_H$ (M**) Flag | Flag byte no.: 0 to 255 | - - - |
| $44_H$ (D**) Data | Data block no.: 2 to 255 | Data word no.: 0 to 255 |

* The start addresses of the memory areas are specified here for the ASCII parameter set and the Send and Receive mailboxes.

** ASCII coded data ID

## 12.4.6 Sample Program for ASCII Driver

Functional sequence of the sample program:
The sample program generates a log for output to the PT88 printer, starting a printout automatically every two seconds. Proceed as follows:

- Set the DIP switches on the printer

  Base socket (front):

| | |
|---|---|
| 1 | ON |
| 2 | ON |
| 3 | ON |
| 4 | ON |
| 5 | ON |
| 6 | ON |
| 7 | OFF |
| 8 | OFF |
| 9 | ON |
| 10 | ON |

- Set the mode selectors on the SAP-S2 interface adapter module (for TTY interface)

| S1: | | S2: | |
|---|---|---|---|
| 1 | ON | 1 | OFF |
| 2 | ON | 2 | OFF |
| 3 | ON | 3 | ON |
| 4 | ON | 4 | ON |
| 5 | OFF | 5 | OFF |
| 6 | ON | 6 | OFF |
| 7 | OFF | 7 | ON |
| 8 | OFF | 8 | ON |

- Connect the PT88 printer to the CPU 943/944 interface SI 2 with the appropriate cable (see Figure 12-12)

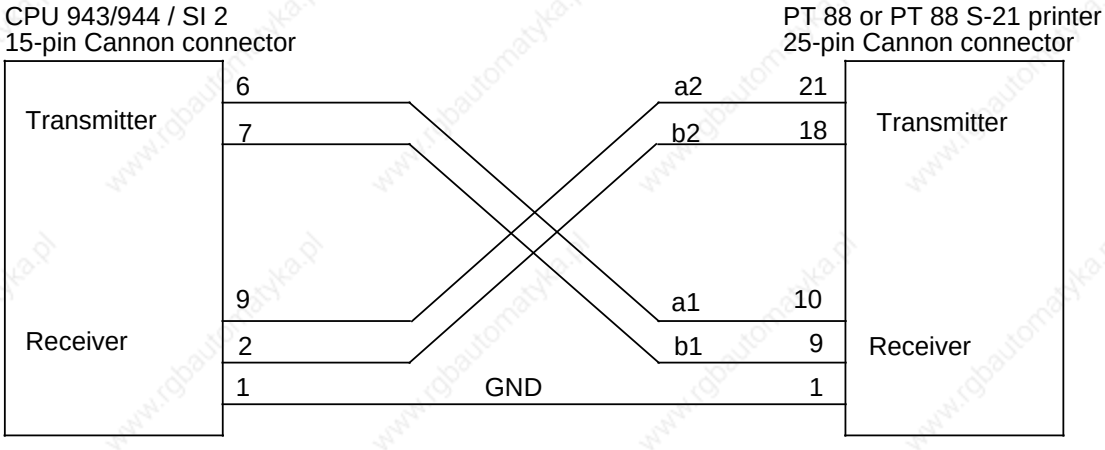



**Figure 12-12. Connector Pin Assignments on the Connecting Cable between the CPU 944/SI 2 and the PT88 or PT88 S-21 Printer (TTY)**

- Position the paper in the printer
- Switch on the printer (in on-line mode)
- Switch on the CPU 943/944 and execute an Overall Reset (CPU operating mode: STOP)
- Enter the program and transfer it to the PLC
- Set the CPU to RUN

The structure of the sample program is shown in Figures 12-13 and 12-14.

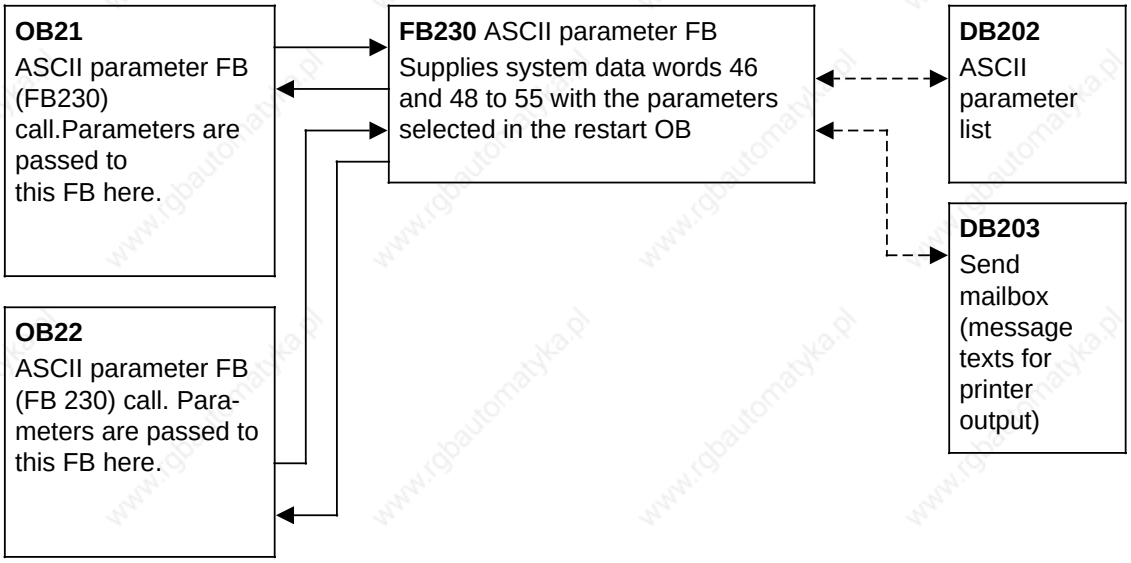



**Figure 12-13. ASCII Driver Program Structure for RESTART**

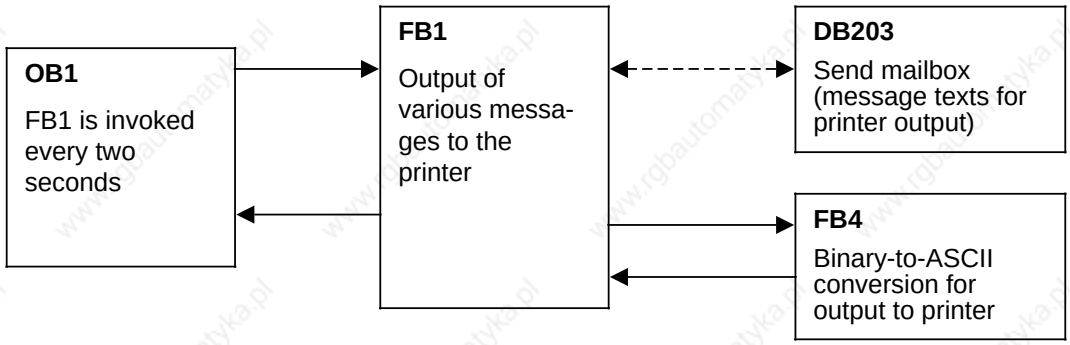



**Figure 12-14. Structure of the Cyclic ASCII Driver Program**

| OB21 STL | Description |
|---|---|
| :<br>:<br>:JU FB 230<br>NAME :ASCII-PA<br>TPAR : KS DB<br>NPAR : KY 202,0<br>TSF : KS DB<br>NSF : KY 203,0<br>TEF : KS XX<br>NEF : KY 0,0<br>TKBS : KS FY<br>NKBS : KY 200,0<br>TKBE : KS FY<br>NKBE : KY 201,0<br>MODE : KF +6<br>:<br>:<br>:BE | ASCII PARAMETER FB CALL<br><br><br>DATA TYPE OF ASCII PARAMETER LIST<br>IS DB202 AND BEGINS WITH DW0.<br>THE SEND MAILBOX IS LOCATED IN<br>DB203 BEGINNING DW0.<br>NOT REQUIRED<br>NOT REQUIRED<br>THE SEND COORDINATION BYTE<br>IS FB200.<br>THE RECEIVE COORDINATION BYTE<br>IS FB201.<br>ASCII DRIVER MODE NUMBER 6 |

| FB230 STL | Description |
|---|---|
| NAME :ASCII-PA | INITIALIZE ASCII PARAMETER LIST |
| DECL :TPAR I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KS |
| DECL :NPAR I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KY |
| DECL :TSMB I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KS |
| DECL :NSMB I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KY |
| DECL :TRMB I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KS |
| DECL :NRMB I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KY |
| DECL :TCBS I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KS |
| DECL :NCBS I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KY |
| DECL :TCBR I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KS |
| DECL :NCBR I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KY |
| DECL :MODE I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KF |

| FB230 STL (Continued) | Description |
|---|---|
| : | FB 230 (CONTINUED) |
| :L KH 0100 | CHANGE ID FOR SI 2 TO ASCII |
| :T FW 200 | |
| :L BS 47 | CURRENT ENTRY CONTAINED IN SYSTEM DATA WORD 47 |
| | |
| :T FW 202 | |
| :LW =TPAR | DATA TYPE OF PARAMETER LIST |
| :T FW 204 | |
| :LW =NPAR | FY OR DB NUMBER OF PARAM. LIST |
| :T FW 205 | |
| :LW =TSMB | DATA TYPE OF SEND MAILBOX |
| :T FW 207 | |
| :LW =NSMB | FY OR DB NUMBER OF SEND MAILBOX |
| :T FW 208 | |
| :LW =TRMB | DATA TYPE OF RECEIVE MAILBOX |
| :T FW 210 | |
| :LW =NRMB | FY OR DB NUMBER OF RECEIVE MAILBOX |
| :T FW 211 | |
| :LW =TCBS | DATA TYPE OF THE CBS |
| :T FW 213 | |
| :LW =NCBS | FY OR DB NUMBER OF THE CBS |
| :T FW 214 | |
| :LW =TCBR | DATA TYPE OF THE CBR |
| :T FW 216 | |
| :LW =NCBR | FY OR DB NUMBER OF THE CBR |
| :T FW 217 | |
| :LW =MODE | DRIVER MODE NUMBER |
| :T FY 219 | |
| :L KH EEDB | ABS. ADDRESS OF FY 219 |
| :L KH EA6F | ADDR. OF SYS. DATA WORD 55 (LOW BYTE) |
| :TNB 20 | |
| :L KH 0000 | RESET SPECIFIED FLAG AREA |
| :T FY 200 | |
| :T FY 202 | |
| :T FY 204 | |
| :T FY 206 | |
| :T FY 208 | |
| :T FY 210 | |
| :T FY 212 | |
| :T FY 214 | |
| :T FY 216 | |
| :T FY 218 | |
| :T FY 220 | |
| :BE | |

| OB1 STL | Description |
|---|---|
| :<br>:AN F 0.0<br>:L KT 200.0<br>:SD T 0<br>:A T 0<br>:= F 0.0<br>:<br>:JC FB 1<br>NAME :DRUCKEN<br>:<br>:BE | <br><br><br><br><br><br>CALL FB 1 EVERY 2 SECONDS |

Sample function block FB1 is used to print out message texts stored in send data block DB203. Output to printer is initiated each time the function block is invoked and the send trigger bit (CBS bit 7) reset.
Each time FB1 is invoked, the number output in the message text is incremented by 1. Function block FB4 converts the message number from binary to ASCII.

| FB1 STL | Description |
|---|---|
| NAME :DRUCKEN | MESSAGE OUTPUT FB |
| :C DB 203 | CALL SEND MAILBOX DB |
| : | |
| :A F 200.7 | CBS BIT: "SEND" (PRINT) |
| :JC =ENDE | |
| : | |
| :L FW 202 | INCREMENT MESSAGE OUTPUT NUMBER |
| :ADD KF +1 | FOR EXAMPLE BY 1 |
| :T FW 202 | |
| : | |
| :JU FB 4 | CALL CONVERSION FB |
| NAME :DU>ASCII | |
| DUAL : FW 202 | SOURCE NUMBER (BINARY) |
| A-TH : DW 21 | ASCII REPRES. T/H (DATA WORDS TO BE UPDATED |
| A-ZE : DW 22 | ASCII REPRES. Z/E IN SEND DB) |
| : | |
| :L FW 204 | INCREMENT ERROR TEXT NUMBER |
| :ADD KF +2 | FOR EXAMPLE BY 2 |
| :T FW 204 | |
| : | |

| FB1 STL (Continued) | Description |
|---|---|
| :JU FB 4 | CALL CONVERSION FB |
| NAME :DU>ASCII | |
| DUAL : FW 204 | |
| A-TH : DW 45 | DATA WORDS TO BE UPDATED |
| A-ZE : DW 46 | IN SEND DB |
| : | |
| :AN F 200.7 | CBS BIT 7 |
| :S F 200.7 | INITIATE PRINTING |
| : | |
| ENDE :BE | |

| FB4 STL | Description |
|---|---|
| NAME :DU>ASCII | CONVERT BINARY NUMBER TO ASCII NUMBER |
| DECL :DUAL I/Q/D/B/T/C: I BI/BY/W/D: W | |
| DECL :A-TH I/Q/D/B/T/C: Q BI/BY/W/D: W | |
| DECL :A-ZE I/Q/D/B/T/C: Q BI/BY/W/D: W | |
| :L KB 0 | CLEAR AUXILIARY REG. |
| :T FW 240 | |
| :T FW 242 | |
| :T FW 244 | REMAINDER REG. |
| : | |
| :L =DUAL | LOAD BINARY NUMBER (VAL. RANGE 0-9999) |
| :L KF +9999 | |
| :>F | |
| :BEC | |
| :TAK | |
| SUBT :L KF +1000 | EVALUATE THOUSANDS PLACE |
| :>=F | |
| :JC =TAUS | JUMP TO PROCESS THOUSANDS PLACE |
| :TAK | |
| SUBH :L KF +100 | EVALUATE HUNDREDS PLACE |
| :>=F | |
| :JC =HUND | JUMP TO PROCESS HUNDREDS PLACE |
| :TAK | |
| SUBZ :L KF +10 | EVALUATE TENS PLACE |
| :>=F | |
| :JC =ZEHN | JUMP TO PROCESS TENS PLACE |
| :JU =EINE | JUMP TO PROCESS ONES PLACE |
| : | |
| TAUS :-F | |
| :T FW 244 | |
| :L FY 240 | |
| :ADD KF +1 | |
| :T FY 240 | INCREMENT COUNTING REG. FOR THOUSANDS |
| :TAK | |

| FB4 STL (Continued) | Description |
|---|---|
| :JU =SUBT | JUMP TO PROCESS THOUSANDS PLACE |
| : | |
| HUND :-F | |
| :T FW 244 | |
| :L FY 241 | |
| :ADD KF +1 | |
| :T FY 241 | INCREMENT COUNTING REG. FOR HUNDREDS |
| :TAK | |
| :JU =SUBH | JUMP TO PROCESS HUNDREDS PLACE |
| : | |
| ZEHN :-F | |
| :T FW 244 | |
| :L FY 242 | |
| :ADD KF +1 | |
| :T FY 242 | INCREMENT COUNTING REG. FOR TENS |
| :TAK | |
| :JU =SUBZ | JUMP TO PROCESS TENS PLACE |
| : | |
| EINE :TAK | |
| :T FY 243 | DESCRIBE COUNTING REG. FOR ONES |
| : | |
| :L KH 3030 | |
| :L FW 240 | |
| :OW | |
| :T =A-TH | |
| :TAK | |
| :L FW 242 | |
| :OW | |
| :T =A-ZE | |
| : | |
| :BE | |

ASCII Driver Parameter Data Block DB202 for Sample Program

| DB202 STL | Description |
|---|---|
| 0: KF = +00008; | Baud rate: 8=9600 baud |
| 1: KF = +00000; | Parity: 0=even parity |
| 2: KF = +00007; | Bits/char.: 7=7 bits |
| 3: KH = 0000; | Waiting time after CR: (none) |
| 4: KH = 0000; | Waiting time after LF: (none) |
| 5: KH = 0000; | Waiting time after FF: (none) |
| 6: KH = 000A; | Delay time between 2 char.: A = 100 ms |
| 7: KH = 0004; | End-of-text char.: "EOT" |
| 8: KH = 0001; | Suppress LF: NO |
| 9: KF = +00066; | Lines/page: 66 |
| 10: KF = +00000; | Left margin: 0 characters |
| 11: KS =' u'; | PAGE NUMBER AT BOTTOM OF PAGE |
| 12: KH = 1B38; | Spaced print ON |
| 13: KS =' MESSAGE LOG: CPU94' | Header line 1 |

| DB202 STL (Continued) | Description |
|---|---|
| 25: KS ='4-ASCII DRIVERS '; | |
| 33: KH = 1B3C; | Spaced print OFF |
| 34: KH = 0D0A; | CR / LF |
| 35: KS ='========================'; | Header 2 |
| 47: KS ='========================'; | |
| 59: KS ='========================'; | |
| 71: KS ='========'; | |
| 75: KH = 0D0A; | CR / LF |
| 76: KS ='************************'; | Footer 1 |
| 88: KS ='**********<Page>********'; | |
| 100: KS ='************************'; | |
| 112: KS ='********'; | |
| 116: KH = 0D0A; | CR / LF |
| 117: KS ='  Exam'; | Footer 2 |
| 129: KS ='ple for CPU944 ASCII dri'; | |
| 141: KS ='ver interface  '; | |
| 151: KH = 0D0A; | CR / LF |
| 152: | |

Send Data Block DB203 for Sample Printer Output Program

| DB203 STL | Description |
|---|---|
| 0: KH = 0A0D; | Control char.: LF / CR |
| 1: KH = 1B5B; | Activate control char. for |
| 2: KH = 3477; | pitch 1/17 |
| 3: KS ='  Process mess' | Message text |
| 15: KS ='age NO.: ': | |
| 20: KH = 1B30; | Control char.:Underline ON |
| 21: KS ='0000'; | Text message number (is used by FB4) |
| 23: KH = 1B39; | Control char.:Underline OFF |
| 24: KS ='  ***  >'; | Message text |
| 28: KH = 1B30; | Control char.:Underline ON |
| 29: KS ='  C A U T I O N  B U R N'; | Message text |
| 41: KS =' E R  0000  H A S F A '; | Message text and message number |
| 53: KS ='I L E D !  '; | Message text |
| 60: KH = 1B39; | Control char.:Underline OFF |
| 61: KS ='< '; | Message text |
| 62: KH = 200D; | SPACE and CR |
| 63: KH = 1B5B; | Activate control char. |
| 64: KH = 3177; | for pitch 1/10 |
| 65: KH = 0A04; | End-of-text char. is EOT (see PAR-DB 202) |
| 66: KH = 0000; | |
| 67: | |

## 12.5 Communications Link Using the 3964/3964R Communications Protocol (for CPU 944 with Two Serial Ports Only*)

A communications link enables data interchange between two programmable controllers (two CPUs) or between a programmable controller and a remote node (with 3965/3964R line procedure). A communications link is possible over interface SI 2 only.
The CPU's application program initiates data interchange, which is then controlled by the 3964 (3964R) line procedure. In contrast to the 3964, the 3964R line procedure generates a block check character (BCC) at the end of each frame, and transmits this character together with the frame. The BCC is the vertical parity of all identically-weighted bits in a frame.
The following configurations are feasible (Figure 12-15):

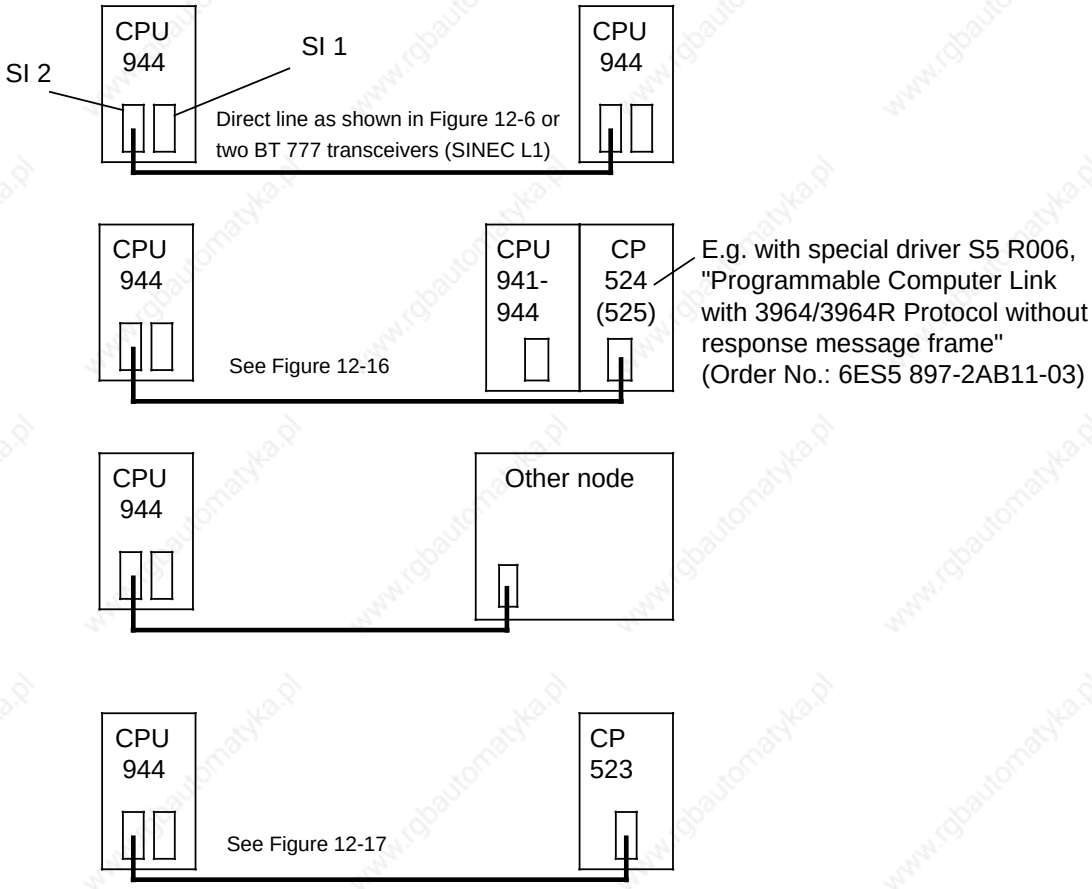



**Figure 12-15. Communications Link between a CPU 944 with Two Serial Ports and a Remote Node**

* With operating system submodule 816-1BB21

**Communications Link between a CPU 944 and a CP 525**

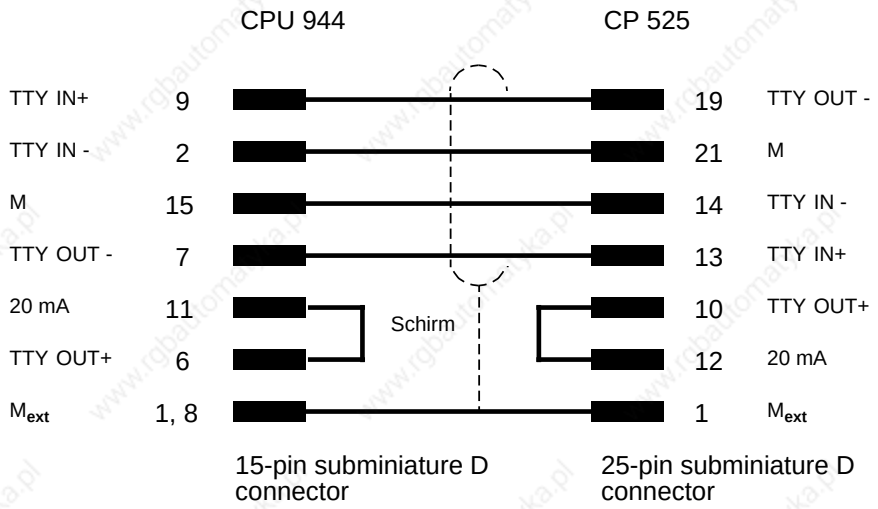



**Figure 12-16. Communications Link between a CPU 944 and a CP 525**

**Communications Link between a CPU 944 and a CP 523**

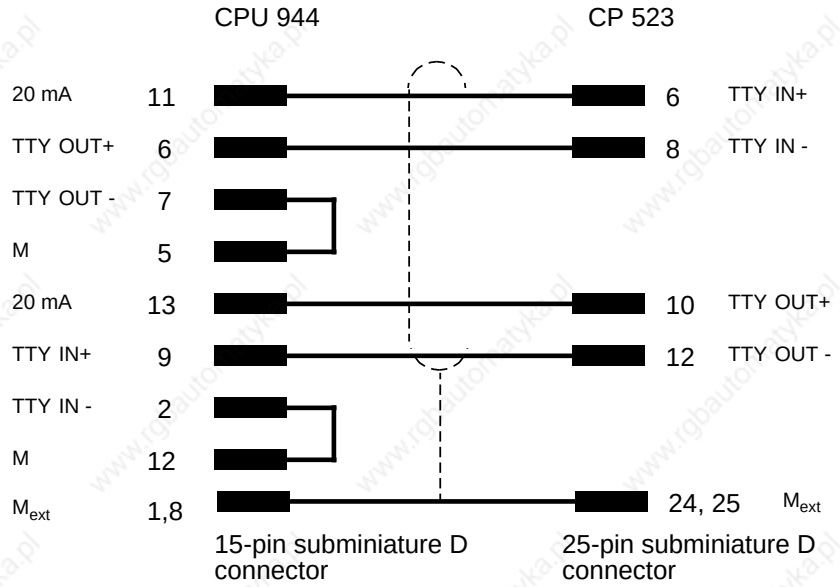



**Figure 12-17. Communications Link between a CPU 944 and a CP 523**

The peers in the link are connected via a direct line (<100 m).

Cable requirements:
- 4-wire
- shielded
- cross-section 0.14 mm$^2$ (26 AWG)

We recommend SIMATIC cable 6ES5 707-1AA00.

### 12.5.1 Data Interchange over the SI 2 Interface

The data to be transferred must be entered in an area of memory designated as the "Send mailbox". Conversely, the data to be received requires a "Receive mailbox", and an area in memory must therefore also be designated for this purpose (detailed information is presented in the next section). The data is stored temporarily in interface SI 2's input or output buffer. Figure 12-18 illustrates the procedures involved in data interchange.

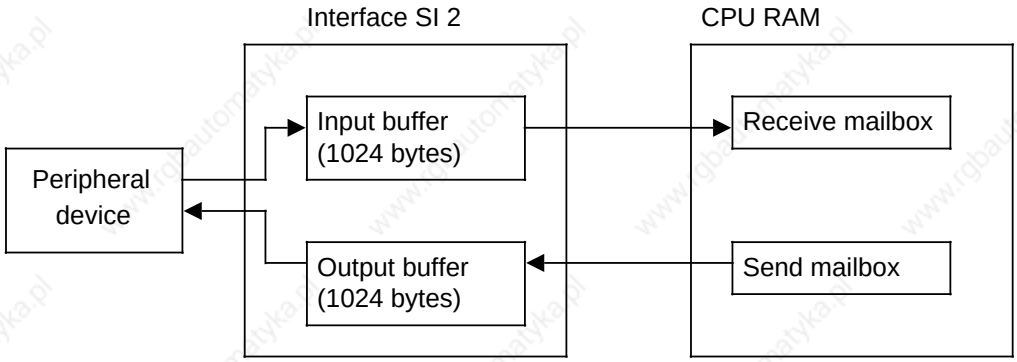



**Figure 12-18. Data Interchange over the SI 2 Interface**

**Initializing a Communications Link**

The 3964/3964R line procedure requires certain information which must be made available in specific system data words.
This includes:

- The location of the data to be transferred in CPU RAM (the so-called "Send mailbox")
- The location of the receive data in CPU RAM (the so-called "Receive mailbox")
- The location of the "Coordination byte, Send" (CBS) and the "Coordination byte, Receive" (CBR) in CPU RAM. These bytes initiate sending/receiving, and also contain an error code should a transmission error occur.
- The location of the parameter list in CPU RAM (this list contains parameters such as the baud rate, parity and the like).

The line procedure also requires the following information, which again must be made available in system data words:

- Mode number (data transmission mode, 3964 or 3964R line procedure)
- Number of the driver for the 3964/3964R line procedure (referred to from now on as the communications link).

The Send mailbox, the Receive mailbox, the Send coordination byte (CBS), the Receive coordination byte (CBR) and the parameter list may be located in either the flag area or in a data block.

You must store the position of the send and receive mailboxes, CBS, CBR, the parameter set and mode number in the user program in system data words 48 to 55, e.g. with the T RS operation. See Table 12-14 for the precise assignment. In addition, the computer link must be activated by entering the driver number in SD 46.

**Table 12-14. Parameter Block for a Communications Link**

| System Data Word | High-Order Byte | Low-Order Byte | Absolute Address |
|---|---|---|---|
| SD 48 | Parameter set<br>Data identifier[1] | Parameter set<br>Flag byte or DB number[2] | EA60 |
| SD 49 | Parameter set<br>Data word number[3] | Send mailbox<br>Data identifier[1] | EA62 |
| SD 50 | Send mailbox<br>Flag byte or DB number[2] | Send mailbox<br>Data word number[3] | EA64 |
| SD 51 | Receive mailbox<br>Data identifier[1] | Receive mailbox<br>Flag byte or DB number[2] | EA66 |
| SD 52 | Receive mailbox<br>Data word number[3] | CBS<br>Data identifier[1] | EA68 |
| SD 53 | CBS<br>Flag byte or DB number[2] | CBS<br>Data word number[3] | EA6A |
| SD 54 | CBR<br>Data identifier[1] | CBR<br>Flag byte or DB number[2] | EA6C |
| SD 55 | CBR<br>Data word number[3] | Mode number | EA6E |

1 $4D_H$ (KH) or "M" (KS) for the flag area, $44_H$ (KH) or "D" (KS) for a data block
2 Flag byte number 0 to 255 or data block number 2 to 255
3 Only when the parameter set is located in a data block, otherwise irrelevant

## 12.5.2 Assigning a Mode Number (System Data Word 55, $EA6E_H$)

Data can be transmitted in one of two modes. The mode selected, that is to say, its code number, must be entered in the low-order byte of system data word 55 (also see Table 12-14).

Refer to Table 12-15 for detailed information regarding the mode.

**Table 12-15. Meaning of the Mode Number**

| Mode | Description |
|---|---|
| 1 | ***No*** block check character (BCC) is transmitted at the end of a frame (3964) |
| 2 | A block check character (BCC) is transmitted at the end of each frame (3964R) |

### 12.5.3 Assigning the Driver Number for a Communications Link

The number of the driver for the communications link is entered in system data word 46 ($EA5C_H$), thus activating the link.

> **Note**
>
> No other functions (e.g. PG/OP) are possible once the communications link has been activated.

The operating system also enters an error code in system data word SD 46 if the specified driver is not available or if coordination bytes have not been defined. Table 12-16 shows the contents of system data word SD 46.

**Table 12-16. System Data Word SD 46**

| Byte | Contents | Description |
|---|---|---|
| High-Order Byte | $00_H$* | Same communications capabilities as the CPU 943 |
| | $02_H$ | Driver for communications link 3964 (R) |
| Low-Order Byte | | **Error Codes** |
| | $01_H$ | No driver for the communications link |
| | $10_H$ | No CBS |
| | $20_H$ | No CBR |
| | $40_H$ | No CBS and no CBR |

* Default value

### 12.5.4 Transmission

Control characters and useful data are transmitted over the interface in bit-serial mode.
When mode 2 is specified in system data word 55, a block check character (BCC) is transmitted at the end of each frame. The BCC itself is protected by the specified parity, and is transferred at the end of each frame. Mode 2 must be entered in system data word 55 to make this possible (see Table 12-14).
Prior to transmission, the data is stored temporarily in a 1024-byte output buffer. An error is flagged (see Table 12-19) if the output buffer does not have sufficient capacity to accommodate the data to be transferred.

The receive data is entered temporarily in a 1024-byte input buffer before being forwarded to the CPU's Receive mailbox via the application program.

### Sending/Receiving with the 3964/3964R Line Procedure in Detail

#### Connection Buildup

The 3964(R) line procedure executes the following steps automatically.

When there is no Send order to process, the 3964(R) driver waits for the peer in the link to establish a connection.

STX is a control character ($02_H$) which initiates connection buildup.

| Possible Responses from the Receiver | Response (from the Transmitter's Point of View) |
|---|---|
| Receiver acknowledges with DLE ($10_H$) prior to time-out (QVZ). | Connection buildup was successful; the transmitter sends the first character from the send buffer.<br>(QVZ: Word 5 in the parameter list) |
| Receiver acknowledges with a character other than DLE or STX prior to time-out (QVZ)<br>or<br>receiver does not acknowledge prior to time-out | Connection buildup was initially unsuccessful; the transmitter makes another attempt to establish a connection. (Number of attempts to establish a connection: Word 7 in the parameter list). If the last attempt also proves unsuccessful, the transmitter enters a code in the Send coordination byte (CBS). |
| Receiver transmits an STX control character prior to time-out (QVZ) | Initiation conflict, i.e. both peers in the link want to transmit. The peer with the lower priority sends DLE, thus enabling the higher-priority node to transmit first; the lower-priority node then transmits its data. The two peers must never have the same priority!<br>(Priority: Word 3 in the parameter set) |

**Sending and Receiving Frames**

- Each character whose value is $10_H$ (DLE) is transmitted twice in succession so that the receiver does not interpret it as the control character for connection buildup. The receiver enters only one of the two characters in its Receive buffer.

- The receiver monitors the time that elapses between transmission of two consecutive characters. If it exceeds the specified character delay time (ZVZ), the receiver sends a NAK and waits the amount of time defined in word 6 of the parameter list for the frame to be retransmitted. (Character delay time: Word 3 in the parameter list).

- The following occurs when the receiver's Receive buffer is full before the transmitter has initiated a connection cleardown:
  - receiving continues until the connection has been cleared down
  - the receiver subsequently transmits the NAK control character
  - the error is flagged in the CBR.

- If the receiver sends a NAK character to the transmitter while transmission is in progress, the transmitter aborts the data transfer and retransmits the entire frame, beginning with the first character.

- If the receiver sends a character other than NAK while transmission is in progress, the transmitter ignores it and continues its transmission

- The receiver reacts to a transmission error (character lost, bad frame, parity error) as follows:
  - Reception continues until the connection is cleared down
  - NAK is then transmitted
  - If an attempt to transmit is still possible (word 8 of the parameter list), the receiver waits for the frame to be retried. How long the receiver waits depends on the frame delay time (word 6 in the parameter list).

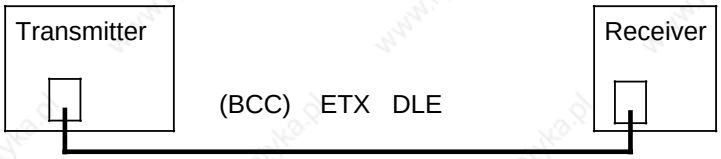
  The receiver aborts transmission and reports an error in CBR
  - if the data block could not be received at the last send attempt
    or
  - if the sender does not start sending within the block waiting time.

- The "BREAK" signal causes the transmitter to
  - abort the current transmission
  - send NAK
  - flag an error in the CBS.

- If a message frame is not accepted (no positive acknowledgement) after the set number of tries to erect or tries to send, the sender responds by sending a NAK.

### Connection Cleardown

When all characters in the Send buffer have been transmitted, the transmitter initiates connection cleardown by transmitting in succession the control characters DLE ($10_H$), ETX ($03_H$) and, if specified, BCC.




| Possible Reactions from the Receiver | Explanation |
|---|---|
| Receiver sends a DLE control character prior to time-out (QVZ) | The frame was received without error and the connection cleared down. |
| Receiver sends a NAK control character or any other character (except DLE !) prior to time-out<br>or<br>the receiver does not send any character prior to time-out | If the specified number of transmission attempts is greater than 1, the frame is retransmitted (number of attempts: word 8 in the parameter list). If the last attempt is unsuccessful, the transmitter aborts the transmission and flags an error in the CBS. |

### Example of Error-Free Transmission

CPU 944 with
3964R line procedure

STX ($02_H$) →
← DLE ($10_H$)
1st char. →
. →
. →
. →
nth char. →
DLE ($10_H$) →
ETX ($03_H$) →
BCC →
← DLE ($10_H$)

**Example of How to Solve an Initiation Conflict**

CPU 944
with 3964R line
procedure
High priority

Low priority

| | |
|---|---|
| | ← STX ($02_H$) |
| STX ($02_H$) → | |
| | ← DLE ($10_H$) |
| 1st char. → | |
| . → | |
| . → | |
| . → | |
| nth char. → | |
| DLE ($10_H$) → | |
| ETX ($03_H$) → | |
| BCC → | |
| | ← DLE ($10_H$) |
| | ← STX ($02_H$) |
| DLE ($10_H$) → | |

**Initializing the Parameter Set**

The parameters required for data interchange are initialized in the parameter set. The location of the parameter set is specified in system data word 48 (or 48 and 49) (see Table 12-14). Refer to Table 12-17 for permissible values and defaults.

> **Note**
>
> The default values are used only when the parameter set was not defined or the parameters could not be interpreted.

**Table 12-17. Parameter Set**

| Word | Description | Value Range | Default |
|---|---|---|---|
| 0 | Baud rate | 2 200 baud<br>3 300 baud<br>4 600 baud<br>5 1200 baud<br>6 2400 baud<br>7 4800 baud<br>8 9600 baud | 8 |
| 1 | Parity | 0 even<br>1 odd<br>2 mark (filler bit high)<br>3 space (filler bit low)<br>4 no check | 0 |
| 2 | Data format* | 0 to 8 | 1 |
| 3 | Priority | 0 low<br>1 high | 1 |
| 4 | Character delay time<br>(The maximum amount of time which may elapse between transmission of two consecutive characters) | 1 - 65535×10 msec. | 22 |
| 5 | Acknowledgement delay-time (time-out)<br>(The time within which a request-to-send or a complete frame (DLE, ETX) must be acknowledged) | 1 - 65535×10 msec. | 200 |
| 6 | Frame delay time<br>(If the character delay time is exceeded, the complete retransmitted frame must arrive in the receiver before the frame delay time is exceeded) | 1 - 65535×10 msec. | 400 |
| 7 | Connection buildup attempts<br>(Maximum number of attempts that may be made to build up a connection) | 1 - 255 | 6 |
| 8 | Number of transmission attempts<br>(Maximum number of attempts that may be made to transmit a block) | 1 - 255 | 6 |

* Meaning of word 2 (data format) see Table 12-18

The parameter set is read at activation of the computer link or after a mode change; data traffic at the interface must have previously been terminated, however (bit 7 in CBR and and bit 7 in CBS=0). The parameter set is also transferred after PLC POWER ON if the computer link had been previously activated.

The parameter settings on the CPU and in the communications partner must be identical to word 3 (priority). The opposite priority must be the default in the communications partner so that an initialization conflict can be resolved.

Please note these time relationships when setting the following:

Character delay time < timeout < block waiting time!

The send or receive process can be initiated when these defaults have been completed.

**Table 12-18. Character Frame and Order of Bits on the Line in the Case of a Computer Link (Depending on Word 2 of the ASCII Parameter Set)**

| Word 2 of the Parameter Set (Data Format) | Char-acter Frame | Parity | Number of Bits per Char-acter | Order of Bits on the Line |
|---|---|---|---|---|
| 0 | 11 bits | 0 to 4* | 7 | 1 start bit, 7 data bits, 1 parity bit, 2 stop bits |
| 1 | 11 bits | 0 to 4* | 8 | 1 start bit, 8 data bits, 1 parity bit, 1 stop bit |
| 2 | 11 bits | Setting irrelevant | 8 | 1 start bit, 8 data bits, 2 stop bits |
| 3 | 10 bits | Setting irrelevant | 7 | 1 start bit, 7 data bits, 2 stop bits |
| 4 | 10 bits | 0 to 4* | 7 | 1 start bit, 7 data bits, 1 parity bit, 1 stop bit |
| 5 | 10 bits | Setting irrelevant | 8 | 1 start bit, 8 data bits, 1 stop bit |
| 6 | - | - | - | - |
| 7 As data format 0 | 11 bits | 0 to 4* | 7 | 1 start bit, 7 data bits, 1 parity bit, 2 stop bits |
| 8 As data format 1 | 11 bits | 0 to 4* | 8 | 1 start bit, 8 data bits, 1 parity bit, 1 stop bit |

* cf. Table 12-17

**Note**

The location of the mailboxes (flag area or data block) must not be changed as long as the "Send/Receive permitted" bits are set.

**Transmitting Data**

- The length of the frame to be transmitted (in bytes) must be entered in the first word of the Send mailbox.

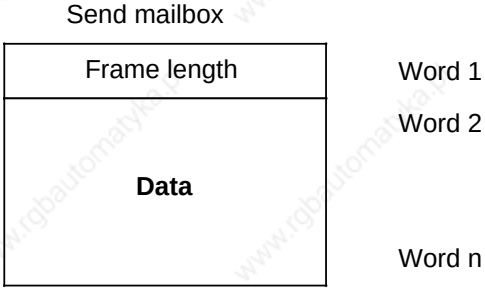



**Figure 12-19. Structure of the Send Mailbox**

- The data to be transmitted must be entered in the remaining words of the Send mailbox.
- Set bit 7 in the CBS (a rising edge triggers the transmission). The driver resets this bit when transmission has been completed.

If data transmission is not possible, bits 0 to 6 of the CBS contain an error code describing the error in more detail. Table 12-19 provides information on the meanings of the various error codes.

**Coordination Byte for Send (CBS)**

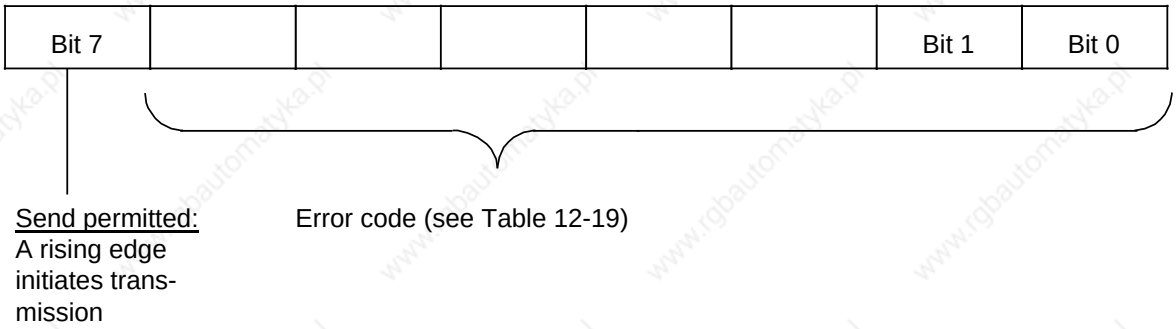



**Figure 12-20. Structure of the CBS**

**Table 12-19. Error Codes in the "Coordination Byte for Send"**

| Error Code | Description | Reaction |
|---|---|---|
| $09_H$ | Negative acknowledgement from receiver during cleardown | Receive data invalid |
| $0B_H$ | Negative acknowledgement from receiver during cleardown | Data transmission not possible |
| $0D_H$ | Parameter assignment error | Data transmission not possible |
| $0F_H$ | Receiver aborted transmission | Receive data invalid |
| $11_H$ | No Send mailbox | Data transmission not possible |
| $13_H$ | Frame exceeds output buffer capacity | |
| $15_H$ | Time-out during connection buildup | |
| $17_H$ | Time-out during connection cleardown | Receive data invalid |
| $19_H$ | Initiation conflict, both peers have high priority | Data transmission not possible |
| $1B_H$ | Break | Transmission is aborted |
| $1D_H$ | Initiation conflict, both peers have low priority | Data transmission not possible |

**Receiving Data**

Receive data is automatically entered in interface SI 2's input buffer (buffer size: 1024 bytes) if the buffer can accommodate it. If it cannot, an error is flagged in the CBR (see Table 12-20).
Bit 7 must be set in the CBR via the application program in order for this data to be forwarded to the Receive mailbox. The driver automatically enters the number of bytes received in the first word of the Receive mailbox, and resets bit 7 in the CBR when the mailbox is full. If the data could not be received without error, an error code is entered in bits 0 to 6 of the CBR; the meanings of the various error codes are listed in Table 12-20.
Since several different errors can occur, the driver assigns them priorities. The CBR always contains the code of the error that had the highest priority when the last attempt to receive the data was made. In Table 12-20, 0 indicates the highest and 6 the lowest priority.

**Coordination Byte for Receive (CBR)**

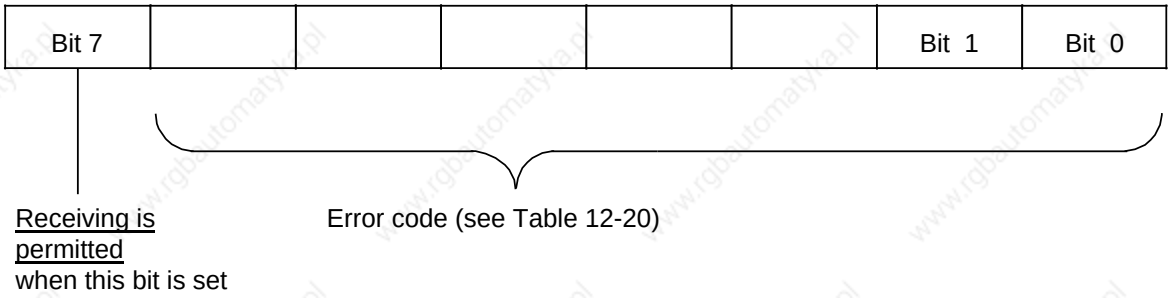



**Figure 12-21. Structure of the CBR**

**Table 12-20. Error Codes in the "Coordination Byte for Receive"**

| Error Code | Description | Priority | Reaction |
|---|---|---|---|
| $03_H$ | Parity error | 5 | Data rejected |
| $05_H$ | Frame length is 0 | 6 | |
| $07_H$ | Input buffer full | 2 | |
| $09_H$ | Too many frames received (more than 100) | 2 | Data valid, subsequent frames were rejected |
| $0B_H$ | Frame longer than Receive mailbox | 0 | Data rejected |
| $0D_H$ | DLE not transmitted twice in succession or no ETX after DLE* | 3 | |
| $0F_H$ | No Receive mailbox | 0 | |
| $11_H$ | STX error: Handshaking did not begin with STX** | 3 | |
| $13_H$ | Character delay time ZVZ exceeded | 4 | |
| $15_H$ | Frame delay time BWZ exceeded | 2 | |
| $17_H$ | Checksum error | 5 | |
| $1B_H$ | Break | 1 | |

* DLE and STX are control characters for connection buildup and connection cleardown (DLE=Data Link Escape, ETC=End of Text). In order that the line procedure be able to distinguish a byte of data from a control character that has the same code (DLE in this case), it automatically doubles the former.
DLE-ETX is a fixed sequence that is required for error-free connection cleardown.

** STX is the control character that establishes a connection with the partner in the communications link (STX=Start of Text).

**Note**

The location of the Send and Receive mailboxes (DB or flag area) must not be changed as long as the "Send/Receive permitted" bits are set.

**Note**

The operating system can set or reset the bits in the coordination bytes after each statement, without regard to the PLC cycle. This means that multiple scanning of a bit in one of these bytes in a program cycle may produce different results (caution is called for in conjunction with edge evaluation!).

### 12.5.5 Sample Program for Transmitting Data

In the restart routine, the parameters for the communications link are passed to system data words 46 and 48 to 55 via a programmable function block (FB220).
The parameters for the communications link are as follows:

- Parameter set in DB202, beginning with DW 0
- Send mailbox in DB203, beginning with DW 0
- Receive mailbox in DB204, beginning with DW 0
- The CBS is flag byte FY 100
- The CBR is flag byte FY 101
- The mode setting is: Mode 2 (with BCC)

The data to be transferred is in data words DW 1 to DW 5 of DB203. The frame length specification must therefore be 10 bytes.

The example describes the program for a remote node, i.e. for a "partner in the link". It can also be used for a remote CPU 944 when that CPU's priority (DB202, DW 3) is changed to "low".

| OB21/OB22 STL | Description |
|---|---|
| :<br>:JU FB 220<br>NAME :PA-3964<br>TPAR : KS DB<br>NPAR : KY 202,0<br>TSMB : KS DB<br>NSMB : KY 203,0<br>TRMB : KS DB<br>NRMB : KY 204,0<br>TCBS : KS FY<br>NCBS : KY 100,0<br>TCBR : KS FY<br>NCBR : KY 101,0<br>MODE : KF +2<br>:<br>:AN F 101.7<br>:S F 101.7<br>:BE | INITIALIZE SYSTEM DATA AREA<br>FOR COMMUNICATIONS LINK<br><br>PARAMETER LIST FOR COMMUNICATIONS LINK<br>IS LOCATED IN DB202 BEGINNING DW0<br>THE SEND MAILBOX IS LOCATED IN<br>DB203 BEGINNING DW0<br>THE RECEIVE MAILBOX IS LOCATED IN<br>DB204 BEGINNING DW0<br>THE COORDINATION BYTE FOR<br>SEND IS FLAG BYTE FY100<br>THE COORDINATION BYTE FOR<br>RECEIVE IS FLAG BYTE FY101<br>MODE NUMBER: 2 (WITH BCC)<br><br>RECEIVE ENABLE |

| FB220 STL | Description |
|---|---|
| NAME :PA-3964 | |
| DECL :TPAR I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KS | |
| DECL :NPAR I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KY | |
| DECL :TSMB I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KS | |
| DECL :NSMB I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KY | |
| DECL :TRMB I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KS | |
| DECL :NRMB I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KY | |
| DECL :TCBS I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KS | |
| DECL :NCBS I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KY | |
| DECL :TCBR I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KS | |
| DECL :NCBR I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KY | |
| DECL :MODE I/Q/D/B/T/C: D KM/KH/KY/KS/KF/KT/KC/KG: KF | |
| : | |
| :L KH 0200 | DRIVER NO. FOR COMM. LINK |
| :T FW 200 | |
| :L RS 47 | RETAIN CURRENT ENTRY IN SYSTEM DATA WORD 47 |
| :T FW 202 | |
| :LW =TPAR | DATA TYPE OF PARAM. SET |
| :T FW 204 | |
| :LW =NPAR | FLAG AREA OR DB NO. OF PARAM. SET |
| :T FW 205 | |
| :LW =TSMB | DATA TYPE OF SEND MAILBOX |
| :T FW 207 | |
| :LW =NSMB | FLAG AREA OR DB NO. OF SEND MAILBOX |
| :T FW 208 | |
| :LW =TRMB | DATA TYPE OF RECEIVE MAILBOX |
| :T FW 210 | |
| :LW =NRMB | FLAG AREA OR DB NO. OF RECEIVE MAILBOX |
| :T FW 211 | |
| :LW =TCBS | DATA TYPE OF CBS |
| :T FW 213 | |
| :LW =NCBS | FLAG AREA OR DB NO. OF CBS |
| :T FW 214 | |
| :LW =TCBR | DATA TYPE OF CBR |
| :T FW 216 | |
| :LW =NCBR | FLAG AREA OR DB NO. OF CBR |
| :T FW 217 | |
| :LW =MODE | MODE |
| :T FY 219 | |
| :L KH EEDB | ABS. ADDR. OF FLAG 219 (SOURCE) |
| :L KH EA6F | ADDR. OF RS 55 (LOW-ORDER BYTE) (DEST.) |
| :TNB 20 | |
| :L KB 0 | RESET FLAG AREA |
| :T FW 200 | |
| :T FW 202 | |

| FB220 STL (Continued) | Description |
|---|---|
| :T FW 204<br>:T FW 206<br>:T FW 208<br>:T FW 210<br>:T FW 212<br>:T FW 214<br>:T FW 216<br>:T FW 218<br>:T FW 220<br>:BE | RESET FLAG AREA |

| OB1 STL | Description |
|---|---|
| :JU FB 1<br>NAME :SEND<br>:<br>:JU FB 2<br>NAME :RECEIVE<br>:BE | SEND<br><br><br>RECEIVE |

| FB1 STL | Description |
|---|---|
| NAME :SEND | |
| :C DB 203 | CALL DB CONTAINING SEND MAILBOX |
| :O F 100.7 | GO TO END WHEN SEND IN PROGRESS |
| :ON I 0.0 | OR WHEN NO SEND REQUEST IS PENDING |
| :BEC | (ENABLE SEND WITH I 0.0) |
| : | |
| :A F 100.0 | ERROR DURING LAST SEND? |
| :JC PB 1 | THEN EVALUATE ERROR IN PB1 |
| : | |
| : | LOAD DATA INTO SEND MAILBOX |
| :L KF +10 | 10 BYTES ARE TO BE TRANSFERRED |
| :T DW 0 | (FIRST WORD IN SEND MAILBOX) |
| : | |
| :L DW 1 | MODIFY END DATA |
| :ADD KF +1 | |
| :T DW 1 | |
| : | |
| :AN F 100.7 | INITIATE SEND |
| :S F 100.7 | |
| :BE | |

| FB2 STL | Description |
|---|---|
| NAME :RECEIVE | |
| :C DB 204 | OPEN DB CONTAINING RECEIVE MAILBOX |
| :A F 101.7 | GO TO END WHEN NO DATA WERE |
| :BEC | RECEIVED. |
| : | |
| :A F 101.0 | RECEIVE ERROR? |
| :JC PB 2 | THEN EVALUATE ERROR IN PB2 |
| : | |
| : | EVALUATE RECEIVE MAILBOX |
| :L DW 0 | EVALUATE NO. OF BYTES RECEIVED |
| :T QW 0 | |
| :L DW 1 | EVALUATE DATA RECEIVED |
| :T QW 2 | |
| : | |
| :AN F 101.7 | RELEASE RECEIVE MAILBOX |
| :S F 101.7 | |
| :BE | |

| DB202 | Description |
|---|---|
| 0: KF = 0008; | BAUD RATE = 9600 BAUD |
| 1: KF = 0000; | EVEN PARITY |
| 2: KF = 0008; | 8 BITS/CHARACTER |
| 3: KF = 0001; | HIGH PRIORITY |
| 4: KF = 0022; | CHAR. DELAY TIME = 220 MS |
| 5: KF = 0200; | ACKNOWLEDGEMENT DELAY TIME = 2 SEC. |
| 6: KF = 0400; | FRAME DELAY TIME = 4 SEC. |
| 7: KF = 0006; | MAX. NO. OF CONN. BUILDUP ATTEMPTS |
| 8: KF = 0006; | MAX. NO. OF ATTEMPTS TO SEND |
| 9: | |

## 13 Integral Real-Time Clock


13.1 Setting the System Data Parameters . . . . . 13- 1
13.2 Structure of the Clock Data Area . . . . . 13- 6
13.3 Structure of the Status Word . . . . . 13- 10
13.4 Battery Backup of the Hardware Clock . . . . . 13- 12
13.5 Programming the Integral Clock . . . . . 13- 13

## Figures


13-1. Control Program and Clock Access to the Clock Data Area . . . . . 13- 6
13-2. Procedure for Reading the Current Date/Time . . . . . 13- 16
13-3. Procedure for Reading the Operating Hours Counter . . . . . 13- 25


## Tables


13-1. System Data Area of the Integral Real-Time Clock . . . . . 13- 2
13-2. Meaning of Bit 0 and Bit 1 in System Data Word 11 . . . . . 13- 3
13-3. Clock Data in the Clock Data Area . . . . . 13- 7
13-4. Clock Data Definition Areas . . . . . 13- 8
13-5. Meaning of the Clock Flags (Bits 0, 1, 2 and 3 of the Status Word) . . . . . 13- 11
13-6. Meaning of Bits 4 and 5 of the Status Word . . . . . 13- 11
13-7. Meaning of the Operating Hours Counter Flags (Bits 8, 9 and 10 of the Status Word) . . . . . 13- 12
13-8. Meaning of the Alarm Clock Flags (Bits 12, 13 and 14 of the Status Word) . 13- 12

# 13 Integral Real-Time Clock
## (only CPU 943/CPU 944 with two serial interfaces)

The integral real-time clock offers the following additional methods of controlling the process:

- Alarm clock function
  e.g. for monitoring the duration of a process
- Operating hours counter
  e.g. for monitoring inspection intervals
- Real-time clock function
  e.g. for establishing the time at which the CPU stopped in the event of a fault

The clock has an accuracy of±2 seconds per day at a temperature of 15 °C. This accuracy changes with temperature according to the following formula:
Temperature dependancy ($T_{amb}$ in °C): t in ms/day=±2s - 3.5 · ($T_{amb}$ - 15)$^2$ ms/day
Example: Tolerance at 40 ° C: ±2 s - 3.5 · (40 - 15)$^2$ ms/day ca. 0 to - 4 s/day.

## 13.1 Setting the System Data Parameters

The hardware clock of the CPU 944 requires a clock data area and a status word to enable use of the clock.
The following information must be stored in system data word 8 to 10:

- The location of the clock data area
- The location of the status word

You can initialize the integral clock via the integral DB1 (parameter block CLP) (see Section 11.3).
The following explains the other method of initializing the integral clock.

#### Initializing the Integral Clock in the System Data Area

You can initialize the clock function in a function block, programmed by you, which is best called via one of the two restart organization blocks OB21 or OB22. The parameters are stored in the relevant system data word in the function block using transfer operations (e.g. "T RS", "TNB").

System data words 8 to 10 are responsible for the location of the clock data area and the status word. These data words determine whether you are dealing with a flag or a data block. They also establish the precise location within the defined area.
The operating system does not implement a standard setting of the system data locations so the clock is switched off under normal circumstances.
Table 13-1 gives details of the meanings of the individual bytes of system data words 8 to 10. System data words 11 and 12 are explained after Table 13-1.

**Table 13-1. System Data Area of the Integral Real-Time Clock**

| Absolute RAM Address | System Data Word | Meaning | Permissible Parameters |
|---|---|---|---|
| EA10 | 8 | Operand area of the clock data | ASCII character: D for DB area<br>F for flag area |
| EA11 | | Initial clock data address<br>Operand area D<br><br>Operand area F | DB number (DB2 to DB255)<br>Flag byte address |
| EA12 | 9 | Initial clock data address<br>(only relevant for operand area D) | Data word number<br>DW 0 to DW 255 |
| EA13 | | Operand area of the status word | ASCII character: D for DB area<br>F for flag area |
| EA14 | 10 | Initial address of the status word<br>Operand area D<br>Operand areas F | DB number (DB2 to DB255)<br>Flag byte address |
| EA15 | | Initial address of the status word<br>(only relevant for operand area D) | Number of the data word<br>DW 0 to DW 255 |
| EA16 | 11 | Start-Up check of the clock block | |
| EA17 | | | |
| EA18 | 12 | Correction value * | - 400 to 0 to+400 |
| EA19 | | | |

** Is only checked and processed once per hour*

**Setting the Clock Parameters**

For safety reasons, a check is made when setting the clock parameters to ensure that the block can be accessed by the operating system and that the clock chip starts up.
Bits 0 and 1 are available in system data word 11 for this purpose. By scanning these bits in the user program, the hardware status can be read out with system statement "L RS 11".

The meaning of these bits is given in Table 13-2.

**Table 13-2. Meaning of Bit 0 and Bit 1 in System Data Word 11**

| System Data Word 11 ($EA16_H$) | | Meaning |
|---|---|---|
| Bit 1 | Bit 0 | |
| 0 | 0 | No second interface |
| 0 | 1 | Clock chip cannot be referenced (defective) |
| 1 | 0 | Clock chip does not start |
| 1 | 1 | Clock chip running correctly |

System data word 11 can be scanned in restart OBs 21 and 22, i.e. you can detect a clock startup failure and output a message accordingly.

Example: Setting the clock parameters during RESTART of the PC (OB21 and OB22)
The clock data are to be stored in data block 2 from data word 0 onwards. The status word is stored in flag word 10. Flag 12.0 is set when the clock does not start up properly.

| OB21 STL | Description |
|---|---|
| :JU FB 101<br>NAME :UHR-INIT<br>TUDA : KS DB<br>NUDA : KY 2,0<br>TUSW : KS FW<br>NUSW : KY 10,0<br>FEHL : F 12.0<br>:<br>:L KM 00000010 00110000<br>:T FW 10<br>:<br>:<br>:<br>:<br>:BE | SET CLOCK PARAMETERS<br><br>CLOCK DATA AREA IS IN DB.<br>HERE: DB2 FROM DW0<br>STATUS WORD OF CLOCK IS FW<br>HERE: FW10<br>ERROR BIT = 1, IF CLOCK<br>NOT CORRECTLY STARTED.<br>PRESET STATUS WORD<br>(HERE E.G.: ENABLE OPERATING HOURS<br>COUNTER, LAST RUN-STOP CHANGE IS<br>STORED, CLOCK TIME UPDATED<br>DURING CPU STOP) |

| FB101 STL | Description |
|---|---|
| NAME :UHR-INIT | UHR INITIALISIEREN |
| DECL :TUDA I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KS |
| DECL :NUDA I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KY |
| DECL :TUSW I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KS |
| DECL :NUSW I/Q/D/B/T/C: D | KM/KH/KY/KS/KF/KT/KC/KG: KY |
| DECL :FEHL I/Q/D/B/T/C: A | BI/BY/W/D: BI |
| :LW =TUDA | OPERAND AREA TYPE |
| :T FW 250 | FOR CLOCK DATA |
| :LW =NUDA | ADDRESS OF CLOCK DATA AREA |
| :T FW 251 | |
| :LW =TUSW | OPERAND AREA TYPE |
| :T FW 253 | FOR STATUS WORD |
| :LW =NUSW | ADDRESS OF STATUS WORD |
| :T FW 254 | |
| : | |
| :L KH EEFF | FINAL ADDRESS OF SOURCE AREA (FB255) |
| :L KH EA15 | FINAL ADDRESS OF DESTINATION AREA (BS10) |
| :TNB 6 | TRANSFER FB 250-255 TO RS 8-10 |
| :L KF +0 | DELETE SCRATCHPAD FLAG |
| :T FW 250 | |
| :T FW 252 | |
| :T FW 254 | |
| :L RS 11 | HAS CLOCK STARTED UP CORRECTLY? |
| :L KH 0003 | |
| :!=F | |
| :RB =FEHL | RESET ERROR BYTE |
| :BEC | |
| :S =FEHL | SET ERROR BYTE |
| :BE | |

| DB2 STL | Description |
|---|---|
| 0: KH = 0003; | --,WEEKDAY //CURRENT TIME |
| 1: KH = 1402; | DAY, MONTH |
| 2: KH = 8908; | YEAR, HOUR + AM/PM BIT |
| 3: KH = 0000; | MINUTE, SECOND |
| 4: KH = 0103; | LEAP YEAR, WEEKDAY //SETTING |
| 5: KH = 1402; | DAY, MONTH |
| 6: KH = 8908; | YEAR, HOUR + AM/PM BIT |
| 7: KH = 0000; | MINUTE, SECOND |
| 8: KH = 0003; | --,WEEKDAY //PROMPT TIME (SETTING) |
| 9: KH = 1402; | DAY, MONTH |
| 10: KH = 0009; | --,HOUR + AM/PM BIT |
| 11: KH = 0000; | MINUTE,SECOND |
| 12: KH = 0000; | --,SECONDS //CURRENT HOURS OF OPERATION |
| 13: KH = 0001; | MINUTES, HOURS |
| 14: KH = 0000; | HOURS X 100, HOURS X 10000 |
| 15: KH = 0000; | --,SECONDS //SETTING HOURS OF OPERATION |
| 16: KH = 0012; | MINUTES, HOURS |
| 17: KH = 0000; | HOURS X 100, HOURS X 10000 |
| 18: KH = 0000; | --,WEEKDAY // CLOCK AFTER STP/RUN |
| 19: KH = 0000; | DAY,MONTH |
| 20: KH = 0000; | YEAR,HOUR |
| 21: KH = 0000; | MINUTE, SECOND |
| 22: | |

**Correction Value**

To compensate for clock inaccuracy due to the effect of temperature, you can enter a correction value in system data word (SD) 12 ($EA18_H$).
The correction value (in seconds) is based on an operating time of 30 days, i.e. if you see that the clock of the CPU 943/944 has lost, say, 20 seconds in 30 days, the correction value is +20.
Internally, the operating system corrects the clock every hour by a value smaller than one second. This ensures that the clock does not "jump" a second (the correction value is read and checked only once per hour). This compensation is unaffected by the mode selected, i.e. it functions in both STOP and RUN mode.

Correction value range: - 400 to 0 to+400 (no correction at "0").
You must specify the correction value in "KF" format.
Following an OVERALL RESET, the default value "0" is in SD 12.

If a nonvalid correction value is used, the operating system sets bit No.15 in SD11; in this case, the correction value is "0".

The time is not corrected on POWER OFF. On POWER-UP, the time correction is updated if the CPU had battery backup during this time. This correction update is implemented only if the powered-down state does not last longer than six months. If the CPU remains off for more than six months, the operating system sets bit 15 in SD11 on POWER-UP and does not correct the time.

## 13.2 Structure of the Clock Data Area

The location of the clock data area must be stored in system data words 8 and 9. Data is always exchanged between the control program and the integrated clock via the clock data area. The integral clock stores current values of clock time, date and operating hours counter in the clock data area (flag area or data block) and, in this same clock data area, the control program stores settings for prompting times and operating hours counters. The control program can only read or write to the clock data area, but can never access the clock direct. Figure 13-1 illustrates this relationship.

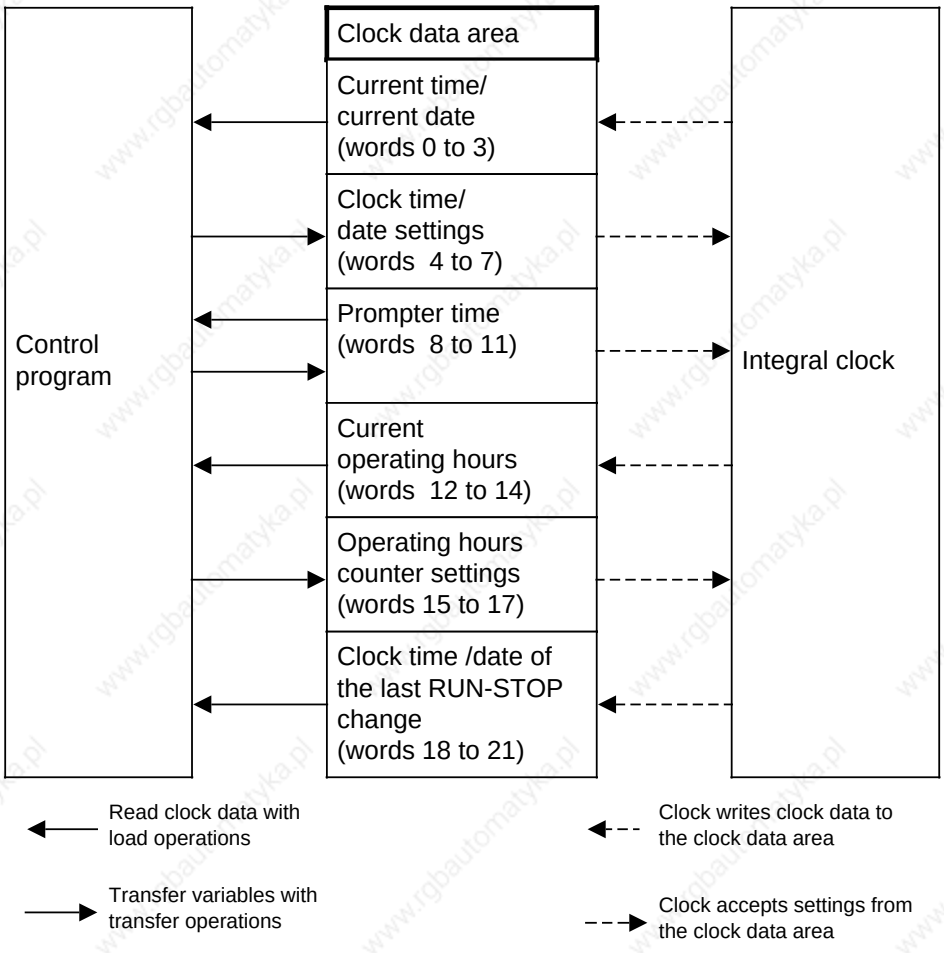



**Figure 13-1. Control Program and Clock Access to the Clock Data Area**

When setting the clock, you need only transfer the data required for implementing the function in question. For example, if you only want to change the data for the clock function, you need not specify the data for the prompter function or for the operationg hours counter.

Table 13-3 gives information on the location of certain clock data within the clock data area, regardless of the memory area selected (DB area or flag area). You will find explanations of the entries in the clock data area following Table 13-3.

**Table 13-3. Clock Data in the Clock Data Area**

| Clock Data Area Word Number | Meaning | Left Word | Right Word |
|---|---|---|---|
| 0 | Current clock time/ current date | --- | Weekday |
| 1 | | Date | Month |
| 2 | | Year | AM/PM (Bit No.7), Hour |
| 3 | | Minute | Second |
| 4 | Time/ Date settings | Leap year | Weekday |
| 5 | | Date | Month |
| 6 | | Year | AM/PM (Bit No.7) , Hour |
| 7 | | Minute | Second |
| 8 | Prompt time | --- | Weekday |
| 9 | | Date | Month |
| 10 | | --- | AM/PM (Bit No.7) , Hour |
| 11 | | Minute | Second |
| 12 | Current operating hours | - | Seconds |
| 13 | | Minutes | Hours |
| 14 | | Hours · 100 | Hours · 10,000 |
| 15 | Operating hours counter settings | - | Seconds |
| 16 | | Minutes | Hours |
| 17 | | Hours · 100 | Hours · 10,000 |
| 18 | Clock time/ date after a RUN/STOP change or POWER OFF (only if bit No. 5 in the status word = 1) | --- | Weekday |
| 19 | | Date | Month |
| 20 | | Year | AM/PM (Bit No.7) , Hour |
| 21 | | Minute | Second |

Please note the following:

- Entries in the clock data area must be in BCD.

- By changing bit No. 1 in the status word, you can select the 12-hour or 24-hour mode for the clock (you will find more details under the heading "Structure of the Status Word").
  The AM/PM flag (0 = AM; 1 = PM) is only of significance if the hardware clock is operating in 12-hour mode. It corresponds to bit 7 of the following words:
  - Word 2
  - Word 6
  - Word 10
  - Word 20.

  In this mode, the hours and the AM/PM flag cannot be set independently of each other when specifying the settings of the clock and the prompting time.

  If an AM/PM flag is set in 24-hour mode, this is recognized when the settings for the clock and prompting time are entered and the relevant error bit is set.

- Settings must lie within the definition ranges given in Table 13-4:

**Table 13-4. Clock Data Definition Areas**

| Variable | Permissible Parameters | Variable | Permissible Parameters |
|---|---|---|---|
| Seconds | 0 to 59 | Day | 1 to 31 |
| Minutes | 0 to 59 | Month | 1 to 12 |
| Hours | in 24-hour mode:<br>0 to 23<br>in 12-hour mode:<br>for AM 1 to 12<br>(12 ≙ 00.00 hours)<br>for PM 81 to 92<br>(92 ≙ noon if AM/PM bit set)<br>0 to 99 if operating hour counter specified | Year | 0 to 99 |
| | | Leap year | 0 to 3<br>0=Leap year is current year<br>1=Leap year was last year<br>2=Leap year was two years ago<br>3=Leap year was three years ago |
| Weekday | 1 to 7<br>1=Sunday<br>2=Monday<br>3=Tuesday<br>4=Wednesday<br>5=Thursday<br>6=Friday<br>7=Saturday | | |

Any other entries will lead to operating system error messages which are flagged in the status word. If the settings are within the definition range, error bits in the status word are reset by the operating system the next time the clock, the prompting time or the operating hours counter is set.
If a setting (prompting time or operating hours counter) is not to be transferred to the clock or if the current value is not to be changed on entry of the setting, enter "FF" (hexadecimal) for this variable.

If the clock data area is located at the end of the individual areas (flags, data block) and if there is insufficient space for the clock data area, only the actual clock data transferred will be accomodated in this area.
Settings outside the range are ignored.

- If the clock data is in the nonretentive flag area, all settings, the operating hours counter and the time of the last RUN/STOP change will be lost after POWER OFF or COLD RESTART!

- Please remember that you can define the location of the clock data area and that the word numbers in Table 13-3 are relative.

  - If your data word area is in a data block and if it does not begin with DW 0 but DW X, you must add the value X to the word number in Table 13-3.

    Example: Your clock data area begins at DW 124. The data for clock time/date are stored from DW 124 to DW 127.

  - If you store the clock data area in the flag area from flag word 0, you must multiply the word number given in Table 13-3 by a factor of 2 in order to obtain the corresponding word address.

    Example: Store the clock data area in the flags operand area from FW 0 onward. The operating hours counter data is stored from address FW 244 onward.

    If your clock data area does not begin at flag word 0, you must add this value.

## 13.3 Structure of the Status Word

The status word can be scanned to detect, for example, errors in the entry of clock settings, or alternatively, specific bits can be changed in the status word to disable or enable transfer or read operations.

The response of the CPU when changing from RUN to STOP or during POWER OFF can be determined with the bits (flags) reserved for this purpose.

- The status word can be located in the flag area or in a data block. The location must be defined in system data words 9 and 10.

- The integral clock runs independently of the mode set.
  Access to the clock data area is dependent on the mode set and the states of bits 4 and 5 of the status word (cf. Table 13-6). You can set or reset these bits with "S" or "R" operations in the control program.
  When monitoring the program with an operator panel (e.g. the OP 396), it is advantageous to have the CPU update the clock (current date) also in STOP mode.

- The "Transfer settings" bits (bits 2, 10 and 14 in the status word) are reset by the operating system if
  - the settings have been transferred
  - the settings have not been transferred because they were outside the permissible range. In this case, the relevant error is set (bits 0, 8 and 12 of the status word).

- The "Transfer settings" bits (bits 2, 10 and 14 of the status word) are not reset by the operating system if
  - the system data for the clock are incorrect or not available
  - the clock data area is too small
  - the clock is defective (hardware error).

- The bits of the status word are divided into
  - clock flags
  - operating system flags
  - operating hours counter flags
  - prompting time flags.

Tables 13-5 to 13-8 contain information on the meaning of the signal states of the flags.

### Clock Flags

**Table 13-5. Meaning of the Clock Flags (Bits 0, 1, 2 and 3 of the Status Word)**

| Bit Number | Signal State | Meaning |
|---|---|---|
| 0 | 1 | Error when entering settings |
| | 0 | No error when entering settings |
| 1 | 1 | 12-hour representation (clock mode) |
| | 0 | 24-hour representation (clock mode) |
| 2 | 1 | Transfer settings |
| | 0 | Do not transfer settings |
| 3 | 1 | Clock time can be read |
| | 0 | Clock time cannot be read |

### Operating System Flags

**Table 13-6. Meaning of Bits 4 and 5 of the Status Word**

| Operating Mode | Bit in the Status Word | Signal State | Meaning |
|---|---|---|---|
| STOP | 4 | 1 | The clock updates only words 0 to 3 in the clock area (current clock time/ current date). The clock can be set with the "FORCE VAR" programmer function. |
| | | 0 | The clock does not update the clock data area. Word 0 to 3 contain the time of the last RUN/STOP change. |
| | 5 | 1 | Words 18 to 21 contain the time of the last RUN/STOP change or the time of the last POWER OFF if bit 4 is also set. |
| | | 0 | Words 18 to 21 are not used. |
| RUN | 4 | 1/0 | The clock updates the clock data area continuously (words 0 to 17). |
| | 5 | 1 | Words 18 to 21 contain the time of the last RUN/STOP change or the time of the last POWER OFF. |
| | | 0 | Words 18 to 21 are not used. |

**Operating Hours Counter Flags**

**Table 13-7. Meaning of the Operating Hours Counter Flags (Bits 8, 9 and 10 of the Status Word)**

| Bit Number | Signal State | Meaning |
|---|---|---|
| 8 | 1 | Error when entering settings |
| | 0 | No error when entering settings |
| 9 | 1 | Enable operating hours counter |
| | 0 | Disable operating hours counter |
| 10 | 1 | Transfer settings |
| | 0 | Do not transfer settings |

**Alarm Clock Flags**

**Table 13-8. Meaning of the Alarm Clock Flags (Bits 12, 13 and 14 of the Status Word)**

| Bit Number | Signal State | Meaning |
|---|---|---|
| 12 | 1 | Error when entering settings |
| | 0 | No error when entering settings |
| 13 | 1 | Set prompting time reached |
| | 0 | Set prompting time not reached |
| 14 | 1 | Transfer settings |
| | 0 | Do not transfer settings |

Bits 6, 7, 11 and 15 are required by the operating system, and cannot be used by the user.

**Scanning the Status Word**

You can scan the individual bits of a data word in a data block using the "P<data word number> <bit number>" operation. Scan the individual bits in the flag area by entering the <byte address> and the <bit number>.

Example: The status word is stored in DW 13. Supposing you want to check to see whether the set prompting time has been reached.
Scanning is initiated with the "B D 13.13" statement.
If the status word is stored in FW 13, the scan operation would be "A F 13.5".

## 13.4 Battery Backup of the Hardware Clock

With battery backup, the clock will continue to operate even after "POWER OFF". If the PC does not have battery backup, the clock will show the settings 01.01.89 12.00.00, Weekday: 1 when the clock is initialized following "POWER ON". The 24-hour mode is set as default. Batteries should therefore only be changed while the power is on, otherwise the clock data will be lost.

## 13.5 Programming the Integral Clock

**Transferring Settings to the Clock**

- Settings are stored in the clock data area with Transfer operations (cf. Table 13-3).
- The AM/PM flag (bit No. 7) is only significant in 12-hour mode.
  Bit 7=1 PM
  Bit 7=0 AM
- Clock data must be transferred in BCD.

> **Note**
>
> The "KC" data format loads a BCD constant into ACCUM 1 and is therefore especially suitable for entering clock settings.

- If a setting is not to be transferred, identify the corresponding byte with the number "$255_D$" or "$FF_H$". The value of this variable in the clock is then retained when the clock is set.
- Once you have transferred the settings to the clock data area, you must set bit 2 of the status word before the clock can accept the clock data.
- Incorrect settings are flagged in the status word by setting bit 0. The clock continues to operate with old values.

**Example:** Transferring new settings (clock time/date) to the clock, using the programmer.

Setting the clock with the following data: Tue 01.03.88; 12.00.00. The status word is assigned to flag word 10 and the clock data is stored in DB2 from data word 0. The settings for the clock data are transferred:

- With the "FORCE VAR" programmer function if the PC is in RUN Mode
- With the "FORCE VAR" programmer function if the PC is in "STOP" mode and status word bit 4=1.

> **Note**
>
> When using the "FORCE VAR" function, you must enter the clock data first and then the status word.

| Operand | Signal States | Meaning |
|---|---|---|
| DB2 | | |
| DW 4 | KH=0003 | Leap year and weekday (TUE) |
| DW 5 | KH=0103 | Date (01) and month (03) |
| DW 6 | KH=8812 | Year (88) and hours (12) |
| DW 7 | KH=0000 | Minute (00) and second (00) |
| MW 10 | KH=0014 | In "STOP" and "RUN" modes:<br>Bit 4=1: The clock area is updated in STOP mode.<br>Bit 2=1: Transfer settings |
| or | | |
| MW 10 | KH=0004 | Only in RUN mode:<br>Bit 4 = 0: clock data is not updated in STOP mode.<br>Bit 2 = 1: Transfer settings |

**Example:** Program for setting clock time and date.

Settings for clock time and date are transferred depending on the signal state at input 12.1. These settings must be transferred to flag bytes 120 and 127 before setting input 12.1 (cf. OB1). Values which are not to be changed must be preset with $FF_H$. Clock mode can be defined with input 14.0 (1=12-hour mode). Input 13.0 is the AM/PM bit for 12-hour mode.
The clock data area is in DB2 from DW 0, and the status word is FW 10.

| OB1 STL | Description |
|---|---|
| :<br>:<br>:<br>:<br>:<br>:A I 12.1<br>:S F 20.0<br>:<br>:JU FB 10<br>NAME :UHR-STEL<br>LPYR : FY 120<br>WDAY : FY 121<br>DATE : FY 122<br>MON : FY 123<br>YEAR : FY 124<br>HOUR : FY 125<br>AMPM : I 13.0<br>MIN : FY 126<br>SEC : FY 127<br>ERR : F 12.1<br>MODE : I 14.0<br>:BE | ================================<br>SETTING CLOCK TIME AND DATE<br>================================<br>LOAD TIME AND DATE VALUES<br>INTO FB 120 TO FB 127 FIRST!<br>CLOCK SETTING TRIGGERED<br>BY SETTING F 20.0 (RESET IN FB 10)<br><br><br><br>LEAP YEAR<br>WEEKDAY<br>DAY<br>MONTH<br>YEAR<br>HOUR<br>AMPM BIT (ONLY IMPORTANT IN 12-HR MODE)<br>MINUTES<br>SECONDS<br>ERROR BIT<br>12-HR MODE: I 14.0 = 1 |

| FB10 STL | Description |
|---|---|
| NAME :UHR-STEL | SETTING THE CLOCK |
| DECL :LPYR I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :WDAY I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :DATE I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :MON I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :YEAR I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :HOUR I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :AMPM I/Q/D/B/T/C: I BI/BY/W/D: BI | |
| DECL :MIN I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :SEC I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :ERR I/Q/D/B/T/C: Q BI/BY/W/D: BI | |
| DECL :MODE I/Q/D/B/T/C: I BI/BY/W/D: BI | |
| :A =MODE | 24-HR MODE = 0, 12-HR MODE = 1 |
| := F 11.1 | (CLOCK MODE STATUS WORD BIT 1) |
| :AN F 20.0 | FLAG IS RESET IF SETTINGS ALREADY |
| :JC =M001 | READ INTO CLOCK DATA AREA |
| :R F 20.0 | |
| : | |
| :C DB 2 | CLOCK DATA AREA |
| :L =LPYR | STORE VALUE FOR LEAP YEAR |
| :T DL 4 | |
| :L =WDAY | STORE VALUE FOR WEEKDAY |
| :T DR 4 | |
| :L =DAY | STORE VALUE FOR DATE |
| :T DL 5 | |
| :L =MON | STORE VALUE FOR MONTH |
| :T DR 5 | |
| :L =YEAR | STORE VALUE FOR YEAR |
| :T DL 6 | |
| :L =HOUR | STORE VALUE FOR HOUR |
| :ON =AMPM | IF 12-HR MODE IS SET, AND |
| :ON =MODE | AMPM BIT = 1 (AFTERNOON), THE |
| :JC =MORN | RELEVANT BIT IN THE CLOCK AREA |
| :L KH 0080 | IS SET |
| :OW | |
| MORN :T DR 6 | |
| :L =MIN | STORE VALUE FOR MINUTES |
| :T DL 7 | |
| :L =SEC | STORE VALUE FOR SECONDS |
| :T DR 7 | |
| :AN F 11.2 | TRANSFER SETTINGS |
| :S F 11.2 | (STATUS WORD IS FW10) |
| :L KT 020.1 | START MONITORING TIME |
| :SV T 10 | |
| M001 :A T 10 | BEC, IF MONITORING TIME |
| :BEC | NOT YET ELAPSED |

| FB10 STL (Continued) | Description |
|---|---|
| :AN F 11.2 | HAVE SETTINGS BEEN TRANSFERRED? |
| :JC =M002 | IF YES, JUMP TO M002 |
| :S =ERR | SET ERROR BIT IF ERRORS |
| :BEU | |
| M002 :AN F 11.0 | ERROR WHEN ENTERING SETTINGS? |
| :RB =ERR | RESET ERROR BIT IF NO |
| :BEC | BEC IF NO ERROR |
| :S =ERR | SET ERROR BIT IF ERROR |
| :BE | |

**Reading the Current Time/Date**

Current data is stored in the first four data words of the clock data area (cf. Table 13-2). This data can be read out from there with Load operations.

To be able to read a correct time, bit 3 of the status word must be set in the control program before the read access. The clock data area is no longer updated when bit 3 is set. You must reset this bit after reading the clock.

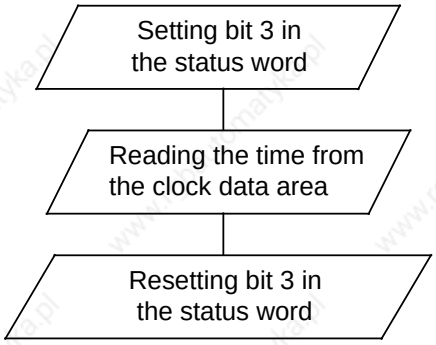



**Figure 13-2. Procedure for Reading the Current Date/Time**

**Example:** Reading the time and the date.

The time is stored in flag bytes 30 to 36 depending on an external event, simulated here by a positive edge at input 12.0. Flag 13.1 indicates which mode the clock is operating in. Flag 13.0 is the AM/PM bit in 12-hour mode.
The clock data area is in DB2 from DW 0 onwards, and the status word is FW 10.

| OB1 STL | Description |
|---|---|
| : | ================================ |
| : | READING TIME AND DATE |
| : | ================================ |
| :A I 12.0 | TIME AND DATE ARE TO BE STORED |
| :AN F 0.1 | IN FY 30 TO FY 36 IN THE CASE OF |
| := F 0.0 | POSITIVE EDGE AT I 12.0 |
| :A I 12.0 | (EXTERNAL EVENT). |
| := F 0.1 | |
| : | |
| :A F 0.0 | EDGE FLAG |
| :JC FB 13 | |
| NAME :UHR-LES | |
| WDAY : FY 30 | WEEKDAY |
| DATE : FY 31 | DAY |
| MON : FY 32 | MONTH |
| YEAR : FY 33 | YEAR |
| HOUR : FY 34 | HOUR |
| AMPM : F 13.0 | M13.0=1, AFTERNOON IN 12-HR MODE |
| MIN : FY 35 | MINUTES |
| SEC : FY 36 | SECONDS |
| MODE : F 13.1 | F13.1=1, IN 12-HR MODE |
| :BE | |

| FB13 STL | Description |
|---|---|
| NAME :UHR-LES | READING THE CLOCK |
| DECL :WDAY I/Q/D/B/T/C: Q BI/BY/W/D: BY | |
| DECL :DAY I/Q/D/B/T/C: Q BI/BY/W/D: BY | |
| DECL :MON I/Q/D/B/T/C: Q BI/BY/W/D: BY | |
| DECL :YEAR I/Q/D/B/T/C: Q BI/BY/W/D: BY | |
| DECL :HOUR I/Q/D/B/T/C: Q BI/BY/W/D: BY | |
| DECL :AMPM I/Q/D/B/T/C: Q BI/BY/W/D: BI | |
| DECL :MIN I/Q/D/B/T/C: Q BI/BY/W/D: BY | |
| DECL :SEC I/Q/D/B/T/C: Q BI/BY/W/D: BY | |
| DECL :MODE I/Q/D/B/T/C: Q BI/BY/W/D: BI | |
| :AN F 11.3 | TIME CAN BE READ |
| :S F 11.3 | (SET BIT 3 IN THE STATUS WORD) |
| :C DB 2 | |
| :L DR 0 | WEEKDAY |
| :T =WDAY | |
| :L DL 1 | DAY |
| :T =DATE | |
| :L DR 1 | MONTH |
| :T =MON | |
| :L DL 2 | YEAR |
| :T =YEAR | |
| :L DR 2 | HOUR |
| :L KH 007F | ERASE AMPM BIT |
| :AW | (ONLY RELEVANT IN 12-HR MODE) |
| :T =HOUR | |
| :P D 2.7 | DISPLAY AMPM BIT |
| := =AMPM | (ONLY RELEVANT IN 12-HR MODE) |
| :L DL 3 | MINUTE |
| :T =MIN | |
| :L DR 3 | SECOND |
| :T =SEC | |
| :A F 11.3 | TIME IS UPDATED AGAIN |
| :R F 11.3 | |
| :A F 11.1 | DISPLAY CLOCK MODE |
| := =MODE | MODE = 1, IN 12-HR MODE |
| :BE | |

**Storing the Current Time/Date After a RUN/STOP Change**

> **Note**
>
> This clock data area is only written to if
> - bit 5 in the status word is set to "1"
> - a RUN/STOP change or a POWER OFF has taken place
> - the necessary memory space is available in the operand area

This enables you to detect a RUN/STOP change or a POWER OFF even if the PC has since gone back to RUN mode. The time and date of the last RUN/STOP change or POWER OFF are in words 18 to 21 (cf.Table 13-3).

If several RUN/STOP changes have occurred before you read out this clock data area, you will only be able to determine the time of the last change.

If you do not have sufficient memory for this clock data area, you cannot use this area or only part of it. This has no other effects.

**Programming the Prompt Function**

**Transferring Settings to the Clock**

- The settings are stored in the clock data area using Transfer operations (cf. Table 13-5).
- The AM/PM flag (bit No. 7) is only significant in 12-hour mode
  Bit 7=1 PM
  Bit 7=0 AM
- The clock data must be transferred in BCD.

> **Note**
>
> The "KC" data format loads a BCD constant into accumulator 1 and is therefore especially suitable for entering settings.

- If you enter the number "$255_D$" or "$FF_H$" in a byte in the prompting time, this byte will be ignored when evaluating "Prompt time reached". This makes it easy to program, for example, a prompt which is repeated daily by entering the value "$255_D$ or $FF_H$" in the "Weekday", "Date" and "Month" variables.
- Transfer of the prompt function settings to the clock is initiated by bit 14 in the status word.
- Incorrect settings are flagged by bit 12 in the status word.

**Prompter Time Sequence**

- Bit 13 in the status word is set after the prompter time has elapsed.
- Bit 13 remains set until you reset it in the control program.
- The prompting time can be read at any time.

> **Warning**
>
> If the prompting time is reached in STOP mode or in POWER OFF, the prompting time cannot be evaluated. It is always deleted on RESTART!

**Example:** Setting and evaluating the prompting time.

In the example program, the settings for the prompting time are transferred as a function of the state of input 12.2. You must transfer the settings to flag bytes 130 and 135 before setting input 12.2. Values that are not to be changed must be preset with $FF_H$.

The clock mode is set with input 14.0. Use input 13.0 to set the AM/PM bit for 12-hour mode.
Flag 13.2 is set when the preset prompting time has been reached. Any errors made when entering the prompter time are flagged in F 12.2.

The clock data is stored in DB2 from DW 0 onwards, and the status word is FW 10.

| OB1 STL | Description |
|---|---|
| : | ========== |
| : | SETTING AND EVALUATING THE PROMPTING TIME |
| : | ========== |
| : | LOAD VALUES INTO FY130 TO FY135 |
| : | FIRST! |
| :A I 12.2 | INITIATE SETTING OF PROMPTING TIME |
| :S F 20.1 | BY SETTING F 21.1 (RESET IN FB11). |
| : | |
| :JU FB 11 | |
| NAME :WECKZ-ST | |
| WDAY : FY 130 | WEEKDAY |
| DATE : FY 131 | DAY |
| MON : FY 132 | MONTH |
| HOUR : FY 133 | HOUR |
| AMPM : I 13.0 | AMPM-BIT (ONLY IMPORTANT IN 12-HR MODE) |
| MIN : FY 134 | MINUTES |
| SEC : FY 135 | SECONDS |
| ERR : F 12.2 | ERROR BIT |
| ALRM : F 13.2 | FLAG FOR PROMPTING TIME REACHED |
| MODE : I 14.0 | 12H-MOD: I 14.0 = 1 |
| :BE | |

| FB11 STL | Description |
|---|---|
| NAME :WECKZ-ST | SETTING THE PROMPTING TIME |
| DECL :WKDAY I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :DATE I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :MON I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :HOUR I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :AMPM I/Q/D/B/T/C: I BI/BY/W/D: BI | |
| DECL :MIN I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :SEC I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :ERR I/Q/D/B/T/C: Q BI/BY/W/D: BI | |
| DECL :ALRM I/Q/D/B/T/C: Q BI/BY/W/D: BI | |
| DECL :MODE I/Q/D/B/T/C: I BI/BY/W/D: BI | |
| :A =MODE | 24-HR MODE = 0, 12-HR MODE = 1 |
| := F 11.1 | (SET CLOCK MODE) |
| :A F 10.5 | DISPLAY PROMPTING TIME REACHED |
| :S =ALRM | (BIT 13 IN THE STATUS WORD) |
| :R F 10.5 | RESET BIT AFTER EVALUATION |
| : | |
| :AN F 20.1 | FLAG IS RESET IF SETTINGS HAVE |
| :JC =M001 | BEEN READ INTO THE CLOCK DATA |
| :R F 20.1 | AREA |
| : | |
| :C DB 2 | CLOCK DATA AREA |
| :L =WKDAY | STORE VALUE FOR WEEKDAY |
| :T DR 8 | |
| :L =DATE | STORE VALUE FOR DATE |
| :T DL 9 | |
| :L =MON | STORE VALUE FOR MONTH |
| :T DR 9 | |

| FB11 STL (Continued) | Description |
|---|---|
| :L =HOUR | STORE VALUE FOR HOUR |
| :ON =AMPM | IF AMPM = 1 (AFTERNOON) AND |
| :ON =MODE | 12-HR MODE ARE SET, THE |
| :JC =MORN | CORRESPONDING BIT ON THE CLOCK |
| :L KH 0080 | DATA AREA WILL BE SET |
| :OW | |
| MORN :T DR 10 | |
| :L =MIN | STORE VALUE FOR MINUTES |
| :T DL 11 | |
| :L =SEC | STORE VALUE FOR SECONDS |
| :T DR 11 | |
| :AN F 10.6 | TRANSFER SETTINGS |
| :S F 10.6 | (BIT 14 IN STATUS WORD FW 10) |
| :L KT 020.1 | START MONITORING TIME |
| :SE T 11 | |
| M001 :A T 11 | BEC IF MONITORING TIME NOT YET |
| :BEC | ELAPSED |
| :AN F 10.6 | HAVE SETTINGS BEEN TRANSFERRED? |
| :JC =M002 | IF YES, JUMP TO M002 |
| :S =ERR | IF ERROR, SET ERROR BIT |
| :BEU | |
| M002 :AN F 10.4 | ERROR WHEN ENTERING SETTINGS? |
| :RB =ERR | IF NO, RESET ERROR BIT |
| :BEC | BEC IF NO ERRORS |
| :S =ERR | IF ERROR, SET ERROR BIT |
| :BE | |

### Programming the Operating Hours Counter

The operating hours counter is enabled with bit 9 of the status word. This allows you to establish, for example, the number of hours a motor has been in operation. The operating hours counter is only active in RUN mode.

### Transferring Settings to the Operating Hours Counter

You can preset the operating hours counter to a specific initial value (e.g. after change of CPU).

- The clock data must be transferred in BCD.

> **Note**
>
> The "KC" data format loads a BCD constant into accumulator 1 and is therefore especially suitable for entering settings.

- If a variable is not to be transferred when you are entering settings for the operating hours counter, identify the relevant byte with the number "$255_D$" or "$FF_H$". The value of this variable in the operating hours counter will then be retained when setting the counter.

- After you have transferred the settings to the clock data area, you must set bit 10 of the status word to have the clock data area accepted by the clock.

- Incorrect settings are flagged by bit 8 in the status word.

**Example:** Setting the operating hours counter

Transferring the settings for the operating hours counter is a function of the state of input 12.3. You must transfer these values to flag bytes 136 to 140 before setting input 12.3 (not implemented in the example program). Values that are not to be changed should be preset with $FF_H$. Incorrect settings are flagged in F 12.3.
The clock data area is in DB2 from DW 0 onwards, and the status word in FW 10.

| OB1 STL | Description |
|---|---|
| : | ================================== |
| : | SETTING THE OPERATING HOURS COUNTER |
| : | ================================== |
| : | LOAD VALUES INTO FY 136 TO FY 140 |
| : | FIRST! |
| :A I 12.3 | INITIATE TRANSFER OF SETTINGS FOR |
| :S F 20.2 | OPERATING HOURS COUNTER BY SETTING |
| : | F 20.2 |
| :JU FB 12 | |
| NAME :BETRST-S | |
| SEC : FY 136 | SECONDS |
| MIN : FY 137 | MINUTES |
| HOUR0: FY 138 | HOURS |
| HOUR2: FY 139 | HOURS X 100 |
| HOUR4: FY 140 | HOURS X 10000 |
| ERR : F 12.3 | ERROR BIT |
| :BE | |

| FB12 STL | Description |
|---|---|
| NAME :BETRST-S | SETTING THE OPERATING HOURS COUNTER |
| DECL :SEC I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :MIN I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :HOUR0 I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :HOUR2 I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :HOUR4 I/Q/D/B/T/C: I BI/BY/W/D: BY | |
| DECL :ERR I/Q/D/B/T/C: Q BI/BY/W/D: BI | |
| :AN F 20.2 | FLAG IS RESET IF SETTINGS HAVE |
| :JC =M001 | ALREADY BEEN READ INTO THE |
| :R F 20.2 | CLOCK DATA AREA |
| : | |
| :C DB 2 | CLOCK DATA AREA |
| :L =SEC | STORE VALUE FOR SECONDS |
| :T DR 15 | |
| :L =MIN | STORE VALUE FOR MINUTES |
| :T DL 16 | |
| :L =HOUR0 | STORE VALUE FOR HOURS |
| :T DR 16 | |
| :L =HOUR2 | STORE VALUE FOR HOURS X 100 |
| :T DL 17 | |
| :L =HOUR4 | STORE VALUE FOR HOURS X 10000 |
| :T DR 17 | |
| :AN F 10.2 | TRANSFER SETTINGS |
| :S F 10.2 | (BIT 10 IN STATUS WORD FW 10) |
| :S F 10.1 | ENABLE OPERATING HOURS COUNTER |
| : | IF NOT ALREADY ENABLED |
| :L KT 020.1 | START MONITORING TIME |
| :SE T 12 | |
| M001 :A T 12 | BEC IF MONITORING TIME NOT YET |
| :BEC | ELAPSED |
| :AN F 10.2 | HAVE SETTINGS BEEN TRANSFERRED? |
| :JC =M002 | IF YES, JUMP TO M002 |
| :S =ERR | IF ERRORS, SET ERROR BIT |
| :BEU | |
| M002 :AN F 10.0 | ERROR WHEN ENTERING SETTINGS? |
| :RB =ERR | IF NO, RESET ERROR BIT |
| :BEC | BEC IF NO ERROR |
| :S =ERR | IF ERROR, SET ERROR BIT |
| :BE | |

**Reading the Current Operating Hours**

The current data is stored in words 12 to 14 of the clock data area. The data can be read from there with Load operations.

Bit 9 in the status word of the control program must be reset before the read access in order to be able to read the operating hours counter correctly. The clock data area is no longer updated when bit 9 is reset. You must set this bit again after reading the clock.

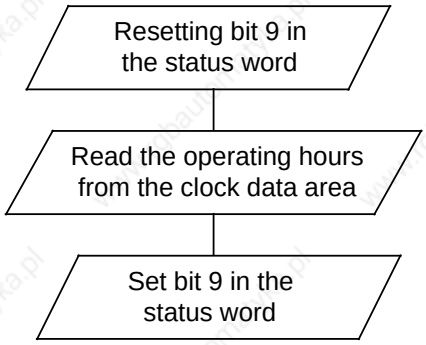



**Figure 13-3. Procedure for Reading the Operating Hours Counter**

**Example:** Reading the operating hours counter

A machine is to be switched off after 300 hours of operation for inspection purposes. Flag 12.4 is set when the machine is switched off. After 300 hours of operation, a jump is made to PB5 to switch the machine off (not programmed in the example).

The clock data area is in DB2 from FW 0 onwards, and the status word is FW 10.

| OB1 STL | Description |
|---|---|
| :JU FB 14<br>NAME :BETR-LES<br>:<br>:BE | EVALUATE OPERATING HOURS COUNTER |

| FB14 STL | Description |
|---|---|
| NAME :BETR-LES | READING THE OPERATING HOURS COUNTER |
| :C DB 2 | ENTER DB IN THE CLOCK DATA. |
| :A F 12.4 | IF AUXILIARY FLAG 12.4 IS SET, |
| :BEC | THE MACHINE IS ALREADY |
| : | OFF. --> BLOCK END |
| :A F 10.1 | DISABLE OPERATING HOURS COUNTER |
| :R F 10.1 | (BIT 9 IN THE STATUS WORD) |
| :L DL 14 | LOAD HOUR VALUE X 100 INTO |
| : | ACCUM 1 |
| :AN F 10.1 | ENABLE OPERATING HOURS COUNTER |
| :S F 10.1 | |
| :L KC 003 | COMPARE WITH 3 (= 300 HOURS) |
| :><F | |
| :BEC | END IF 300 HOURS NOT YET |
| : | REACHED |
| :S F 12.4 | SET AUXILIARY FLAG |
| :JU PB 5 | JUMP TO PB 5 WHEN 300 OPERATING |
| : | HOURS REACHED |
| : | |
| :BE | |

## 14 Reliability, Availability and Safety of Electronic Control Equipment


14.1 Reliability . . . . . 14 - 1
14.1.1 Failure Characteristics of Electronic Devices . . . . . 14 - 2
14.1.2 Reliability of SIMATIC S5 Programmable Controllers and Components . . . . . 14 - 2
14.1.3 Failure Distribution . . . . . 14 - 3

14.2 Availability . . . . . 14 - 4

14.3 Safety . . . . . 14 - 5
14.3.1 Types of Failures . . . . . 14 - 5
14.3.2 Safety Measures . . . . . 14 - 6

14.4 Summary . . . . . 14 - 7

**Figures**


14-1. Failure Characteristics of Electronic Devices ("Bathtub" Curve) . . . . . 14 - 2
14-2. Distribution of Failure Occurences in Installations Incorporating Programmable Controllers . . . . . 14 - 3
14-3. Control of Function "Fx" . . . . . 14 - 5

# 14 Reliability, Availability and Safety of Electronic Control Equipment

The terms reliability, availability and safety of electronic control equipment are not always clear and sometimes even misinterpreted. This can be explained on the one hand by the different failure characteristics of electronic control systems compared with conventional systems. On the other hand, some of the safety regulations have been made considerably more stringent in a number of application areas in the course of the last few years. The following chapter is intended to familiarize the large number of users of SIMATIC electronic control systems with the basics of this problem complex.

The information given is of a predominantly fundamental nature and applies regardless of the type of electronic control system and its manufacturer.

## 14.1 Reliability

Reliability is the capability of an electronic control system to satisfy, over a specified period and within the specified limits (i.e. technical data), the requirements placed upon it by its application.

Despite all the measures taken to prevent failures, there is no such thing as 100 % reliability.

The failure rate is a measure of the reliability:

$$\lambda = \frac{n}{N_0 \times t}$$ and n=Number of failures during time t

$N_0$=Remaining components

### 14.1.1 Failure Characteristics of Electronic Devices

The failure-rate-versus-time curve can be broken down roughly into three periods of time.

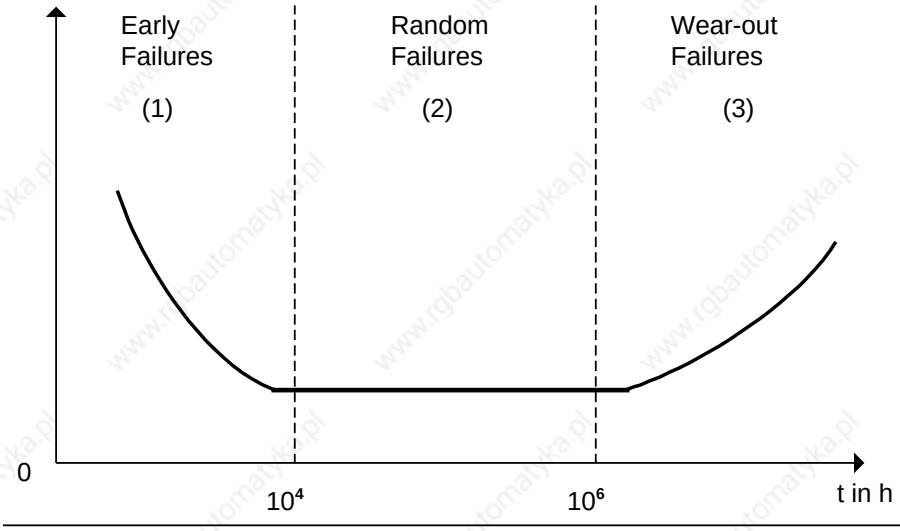



**Figure 14-1. Failure Characteristics of Electronic Devices ("Bathtub" Curve)**

(1) Early failures are caused by material and manufacturing defects and the failure rate falls steeply during the initial period of operation.
(2) The random failure phase is characterized by a constant failure rate. Provided the systems are used in accordance with the specifications, only random failures occur during this period.
This period covers the normal behaviour of system components and is the basis for the calculation of all reliability parameters.
(3) The failure rate increases with time. Wear-out failures become more frequent, indicating that the end of the useful life is approaching. The transition to this phase is gradual. There is no sudden increase in the failure rate.

### 14.1.2 Reliability of SIMATIC S5 Programmable Controllers and Components

A very high degree of reliability can be achieved by taking the following extensive and cost-intensive measures during the development and manufacture of SIMATIC S5 systems:

- The use of high-quality components;
- Worst-case design of all circuits;
- Systematic and computer-controlled testing of all components supplied by subcontractors;
- Burn-in of all LSI circuits (e.g. processors, memories etc.);
- Measures to prevent static charge building up when handling MOS ICs;
- Visual checks at different stages of manufacture;
- In-circuit testing of all components, i.e. computer-aided testing of all components and their interaction with other components in the circuit;
- Continuous heat-run test at elevated ambient temperature over a period of several days;
- Careful computer-controlled final testing;
- Statistical evaluation of all failures during testing to enable the immediate initation of suitable corrective measures.

### 14.1.3 Failure Distribution

Despite the extensive measures described above, one must still reckon with the occurence of failures. Experience has shown that, in installations with programmable controllers, failures can be distributed approximately as follows:

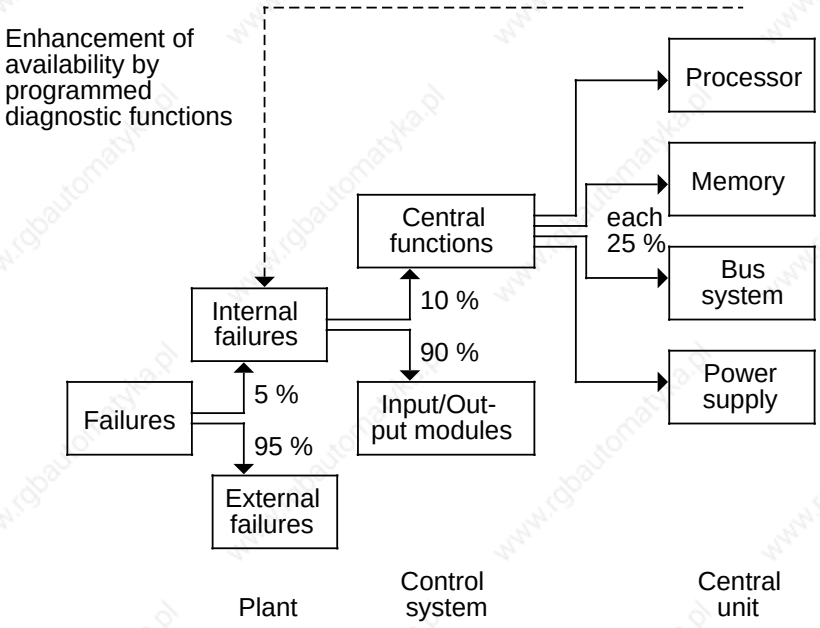



**Figure 14-2. Distribution of Failure Occurrences in Installations Incorporating Programmable Controllers**

Meaning of error distribution:

- Only a small number (approx. 5 %) of failures occur inside the electronic control system. These can be broken down as follows:
  - CPU failures (about 10 %, i.e. only 0.5 % of all failures);
    these failures are evenly divided among the processor, memory, bus system and power supply.
  - I/O module failures (about 90%, i.e. only 4.5 % of all failures)

- The highest number of all failures (about 95 %) occur in the sensors, actuators, drives, cabling etc.

## 14.2 Availability

Availability "V" is the probability of finding a system in a functional state at a specified point in time.

$$V = \frac{MTBF}{MTBF+MTTR}$$

MTBF= Mean Time Between Failures;
MTTR= Mean Time To Repair;

Ideal availability, i.e. V=1, can never be attained owing to the residual failure probability that always exists.

However, it is possible to get near this ideal state by using, for example, voter systems. Such systems include the following:

- Standby sytems
- 2-out-of-3 voter systems
- Multi-channel voter systems with mutual check functions (for maximum safety requirements).

Availability can also be enhanced by reducing the mean time to repair. Such measures include, for instance:

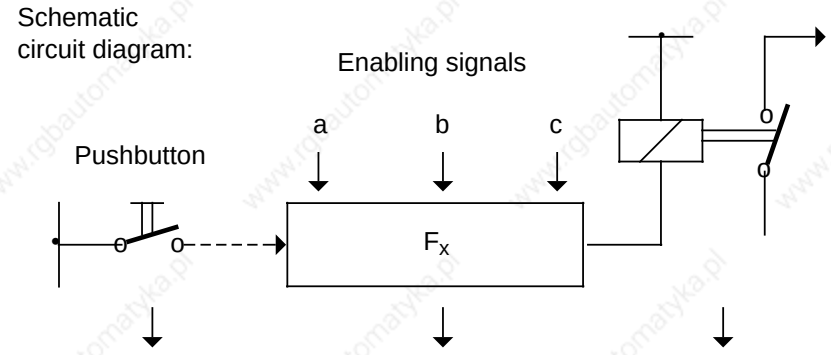
- the stocking of spare parts
- the training of operating personnel
- fault indicators on the devices
- higher memory and software overhead for implementing programmed diagnostic functions.

## 14.3 Safety

### 14.3.1 Types of Failures

The nature of a failure is decided by the effect it has. A distinction is made between active and passive failures, as well as fatal and non-fatal failures.

**Example**: Control of function "$F_x$"




| Input | Control System | Output |
|---|---|---|
| 0<br>1 | No fault | No output command<br>Output command |
| 0<br>1 | Active failure<br>Passive failure | Output command<br>No output command |

**Figure 14-3. Control of Function "Fx"**

Depending on the job a control system has to do, active or passive failures can also be fatal faults.

Examples:

- In a drive control system, an active failure results in the unauthorized starting of the drive.
- In an indicating system, a passive fault can be fatal since it blocks the indication of a dangerous operating state.

In all cases where the occurence of failures can result in severe material damage or even injury to persons, i.e. where the failure may be dangerous or fatal, measures must be taken to enhance the safety of the control system. In this connection, the relevant regulations and specifications must be observed.

## 14.3.2 Safety Measures

**Single-Channel Configurations**

In the case of single-channel programmable controllers, the means available for enhancing safety are limited:

- Programs or parts can be stored and executed more than once.
- Outputs can be monitored per software by parallel feedback to inputs of the same device.
- Diagnostic functions within the programmable control system, which bring the output of the controller into a defined state (generally the FF state) when a failure occurs.

Failure characteristics of electromechanical and electronic control systems:

- Relays and contactors pick up only if a voltage is applied to the coil. With such a control element, therefore, active failures are less probable than passive failures.
- In electronic control systems, however, the probability of both types of failure occurring (active and passive) is approximately equal. The failing of an output transistor, for instance, may cause this transistor to become either continuously non-conducting or continuously conducting.

The safety of electronic control systems can therefore be enhanced as follows.

- All functions not relevant to the safety of the plant are controlled electronically.
- Functions that are relevant to the safety of the plant are implemented with conventional control elements.

**Multi-Channel Configurations**

If the measures taken to improve safety in single-channel control systems are not sufficient to satisfy safety requirements, electronic control systems should be designed as redundant, i.e. multi-channel, systems.

- Two-channel control systems
  Both "channels" monitor each other mutually and the output commands are evaluated on a "1-out-of-2" or "2-out-of-2" basis.
  Typical PLC: S5-115F
  This programmable controller consists of two submits that are identically programmed and operate in clock synchronism; monitoring is implemented via two comparator modules. Failures are displayed and the corresponding safety functions initiated.
- Multi-channel control systems
  Further voter systems (e.g. on the 2-out-of-3 principle) can be implemented by adding further "channels".

## 14.4 Summary

- In electronic control systems, failures of any kind can occur at any point in the system.
- Even when the greatest efforts are made to obtain maximum reliability, the probability of such a failure occurring can never be zero.
- The following is decisive for the effects of such failures: depending on the job a control system has to do, active or passive failures may be fatal or non-fatal.
- When safety requirements are very high, fatal failures must be recognized by taking additional measures and prevented from affecting other parts of the system.
- In the case of single-channel systems, the means available to do this are relatively limited.
  For this reason, safety-oriented functions should generally be implemented outside the electronics by interposing conventional components.
- In order to satisfy safety functions, electronic control systems should be of the multi-channel (redundant) type.
- These fundamental considerations are independent of
  - the type of control systems (hard-wired or programmable)
  - the vendor
  - the country of origin (Europe, US, etc.).

## 15 Technical Specifications


15.1 General Technical Specifications . . . . . 15 - 3

15.2 Description of Modules . . . . . 15 - 5
15.2.1 Mounting Racks (CRs, ERs) . . . . . 15 - 5
15.2.2 Power Supply Modules . . . . . 15 - 9
15.2.3 Central Processing Units . . . . . 15 - 14
15.2.4 Digital Input Modules . . . . . 15 - 20
15.2.5 Digital Output Modules . . . . . 15 - 31
15.2.6 Digital Input/Output Module . . . . . 15 - 45
15.2.7 Analog Input Modules . . . . . 15 - 46
15.2.8 Analog Output Modules . . . . . 15 - 52
15.2.9 Intelligent Input/Output Modules . . . . . 15 - 58
15.2.10 Communications Processors . . . . . 15 - 59
15.2.11 Interface Modules . . . . . 15 - 60
15.2.12 The 313 Watchdog Module . . . . . 15 - 64

15.3 Accessories . . . . . 15. - 65

**Tables**

15-1. Overview of Intelligent Input/Output Modules . . . . . 15- 58
15-2. Overview of Communications Processors . . . . . 15- 59

# 15 Technical Specifications

The following section describes the standards and test values the S5-115U meets and fulfills and the test criteria with which the S5-115U has been tested.

**UL/CSA approvals**

The following approvals exist for the S5-115U:
UL Recognition Mark
Underwriters Laboratories (UL) in accordance with Standard UL 508, File E 116536
CSA Certification Mark
Canadian Standard Association (CSA) in accordance with Standard C 22.2 No. 142, File LR 48323

**CE marking**

Our products meet the requirements and protection objectives of the following EC Directives and comply with the harmonized European standards (EN) published in the Official Gazettes of the European Communities with regard to programmable controllers:

- 89/336/EC "Electromagnetic Compatibility" (EMC Directive)
- 73/23/EC "Electrical Equipment Designed for Use between Certain Voltage Limits" (Low-Voltage Directive)

CE The EC declarations of conformity are held at the disposal of the competent authorities at the address below:

Siemens Aktiengesellschaft
Automation and Drives Group
A&D AS E 14
P.O. Box 1963
D-92209 Amberg
Federal Republic of Germany

**Area of Application**

SIMATIC products have been designed for use in industrial environments.
With individual approval, SIMATIC products can also be used in residential environments (residential, commercial and light industry).
You must acquire the individual approval from the appropriate national authority or testing board.

| Area of Application | Requirements in respect of | |
|---|---|---|
| | Emitted interference | Immunity |
| Industry | EN 50081-2 : 1993 | EN 50082-2 : 1995 |
| Residential | Individual approval | EN 50082-1 : 1992 |

**Oberserving the Installation Guidelines**

S5 modules meet the requirements, if installed and operated in accordance with the Installation Guidelines (see Chapter 3).

**Notes for the machine manufacturer**

The SIMATIC automation system is not a machine in the sense of the EC Directives Machines. Therefore a declaration of conformity with regard to the EC Directive Machines 89/392/EC does not exist for SIMATIC.

The EC Directive Machines 89/392/EC regulates the requirements on a machine. A machine in this sense is a group of interconnected parts or devices (see also EN 292-1, § 3.1).

The SIMATIC is part of the electrical equipment of a machine and must therefore be included in the declaration of conformity procedure by the machine manufacturer.

The standard EN 60204-1 (safety of machines, general requirements for the electrical equipment of machines) applies for the electrical equipment of machines.

The following table is intended to help you with the declaration of conformity and shows which criteria apply to SIMATIC in accordance with EN 60204-1 (June 1993).

| EN 60204-1 | Topic/criterion | Remarks |
|---|---|---|
| § 4 | General requirements | Requirements are fulfilled if the devices are assembled/installed in accordance with the Installation Guidelines.<br>Please also observe the section on CE marking. |
| § 11.2 | Digital input/output interfaces | Requirements are fulfilled. |
| § 12.3 | Programmable equipment | Requirements are fulfilled if the devices are installed in lockable cabinets to protect them from memory modifications through unauthorized persons. |
| § 20.4 | Voltage tests | Requirements are fulfilled. |

## 15.1 General Technical Specifications

**Climatic Environmental Conditions**

**Temperature**

Operation
- Open Design
  air intake temperature (measured at the bottom of the modules) 0 to+55 ° C
- Cabinet design
  (Where cabinet design is concerned, the dissipative heat loss depends on the type of construction, the ambient temperature, and the arrangement of the devices.)
  Air intake temperature (measured at the bottom of the modules) 0 to+55 ° C

Storage/Shipping - 40 to+70 ° C

Temperature change
- operation maximum 10 K/h
- storage/shipping maximum 20 K/h

**Relative Humidity**
- operation 95 % (according to DIN 40040)
- storage/shipping 95 % (noncondensing)

**Atmospheric pressure**
- operation 860 to 1060 hPa [1]
- storage/shipping 660 to 1060 hPa [1]

**Pollutants**
- $SO_2$ 0.5 ppm, (rel. humidity 60 %, noncondensing)
- $H_2S$ 0.1 ppm, (rel. humidity 60 %, noncondensing)

[1] For use under 900 hPa (=1000 m above sea level), check with the manufacturer on the cooling requirements.

**Mechanical Environmental Conditions**

**Vibration***
- tested to IEC 68-2-6

| | |
|---|---|
| 10 Hz f < 57 Hz, | Const. ampl. 0.075 mm |
| 57 Hz f < 150 Hz, | Const. accel. 1 g |
| Mode of vibration | Frequency sweeps with a sweep rate of 1 octave/min |
| Vibration period | 10 frequency sweeps per axis in each of the 3 axes vertical to each other |

- Operation conditions in accordance with IEC 1131-2

**Shock***
- tested to IEC 68-2-27

| | |
|---|---|
| Type of shock | Half sine |
| Strength of shock | 15 g peak value, 11 ms duration |
| Direction of shock | 2 shocks in each of the 3 axes vertical to each other |

**Free-fall**
- tested to IEC 68-2-3
  Height of fall 1 m

* Vibrations and shocks persisting for a prolonged period at the maximum values stated above and continuous shocks should be avoided by taking the appropriate measures.

**Electromagnetic Compatibility (EMC)**
**Noise Immunity**

**Immunity to static electrical discharge**
- tested in accordance with EN 61000-4-2
- Discharge to air 8 kV
- Discharge on contact 4 kV

**Immunity to electromagnetic fields**
- tested in accordance with EN V 50140 (amplitude-modulated HF) 80 to 1000 MHz, 10 V/m, 80 % AM (1 kHz)
- tested in accordance with EN V 50204 (pulse-modulated HF) 900 MHz, 10 V/m, 50 % ED, 200 Hz repetition frequency

**Immunity to fast transient bursts**
- tested in accordance with EN 61000-4-4
- Supply lines for 120/230 V AC 2 kV
- Supply lines for 24 V DC 2 kV
- Signal lines (I/O and bus lines) 2 kV*

**Immunity to high frequency**
- tested in accordance with EN V 50141 0.15 to 80 MHz, 10 V, 80 % AM (1 kHz), Source impedance 150

**Emitted interference**
- tested in accordance with EN 55011
- Emmission of electromagnetic fields: Limit value class A, Group 1
- Emitted interference over supply cable: Limit value class A, Group 1

**Damped Oscillatory Wave Test** according to IEC 255-4
- AC power supply modules 2.5 kV
- DC power supply modules 1 kV
- output 24 V DC 1 kV
- input 115/230 V AC 2.5 kV
- digital input/output modules 2.5 kV
- analog input/output modules 1 kV
- communications interfaces 1 kV

* Signal lines that are not used for process control, for example, connections to external printers: 1 kV

**IEC / VDE**
**Safety Information**

**Degree of Protection** according to IEC 529
- Type IP 20
- Class according to IEC 536

**Isolation Rating**
- between electrically independent circuits **and** with circuits connected by a central grounding point: according to VDE 0160
- between all circuits **and** central grounding point (standard mounting rail): according to VDE 0160

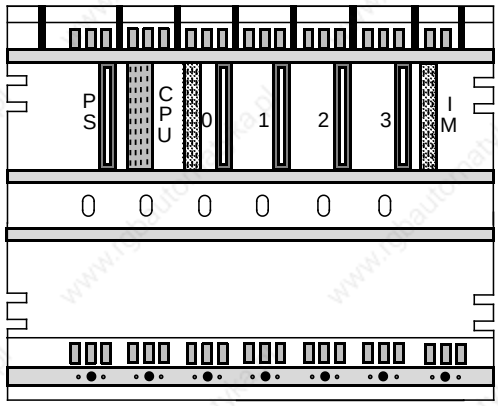
Test Voltage for a rated voltage $V_e$ of the circuits (AC/DC) — Sine, 50 Hz
- $V_e$=0 to 50 V: 500 V DC
- $V_e$=50 to 125 V: 1250 V AC
- $V_e$=125 to 250 V: 1500 V AC

Impulse voltage for a rated voltage $V_e$ of the circuits (AC/DC) — according to IEC 255-4
- $V_e$=0 to 50 V: 1 kV, 1.2/50 µsec.
- $V_e$=50 to 125 V: 1 kV, 1.2/50 µsec.
- $V_e$=125 to 250 V: 3 kV, 1.2/50 µsec.

## 15.2 Description of Modules

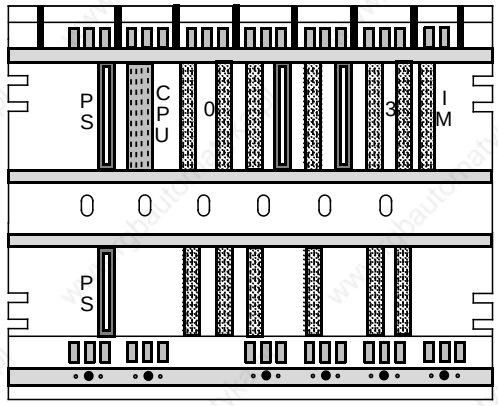
### 15.2.1 Mounting Racks (CRs, ERs)

**Mounting Rack CR 700-0 for Central Controller 0** **(6ES5 700-0LA12)**




| Technical Specifications | | |
|---|---|---|
| Number of input/output modules that can be plugged in | maximum | 4 |
| Number of expansion units that can be connected - central | maximum | 3 |
| Dimensions w x h x d (mm) | | 353 x 303 x 47 |
| Weight | | 4 kg (8.82 lb.) |

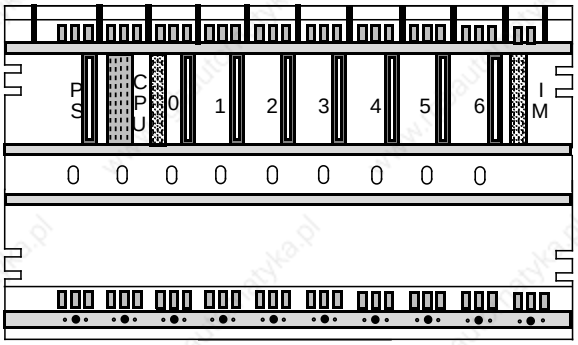
**Mounting Rack CR 700-0 for Central Controller 0** **(6ES5 700-0LB11)**




| Technical Specifications | | |
|---|---|---|
| Number of input/output modules that can be plugged in | maximum | 6 |
| Number of expansion units that can be connected - central | maximum | 3 |
| - distributed * | maximum | 63 |
| Dimensions w x h x d (mm) | | 353 x 303 x 47 |
| Weight | | 4 kg (8.82 lb.) |

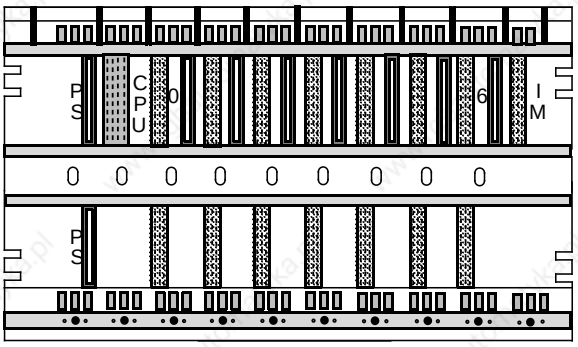
* see Section 3.2.6

## Mounting Rack CR 700-1 for Central Controller 1 (6ES5 700-1LA12)




| Technical Specifications | | |
|---|---|---|
| Number of input/output modules that can be plugged in | maximum | 7 |
| Number of expansion units that can be connected - central | maximum | 3 |
| Dimensions w x h x d (mm) | | 483 x 303 x 47 (18.84 in. x 11.82 in. x 1.83 in.) |
| Weight | | 5 kg (11.02 lb.) |

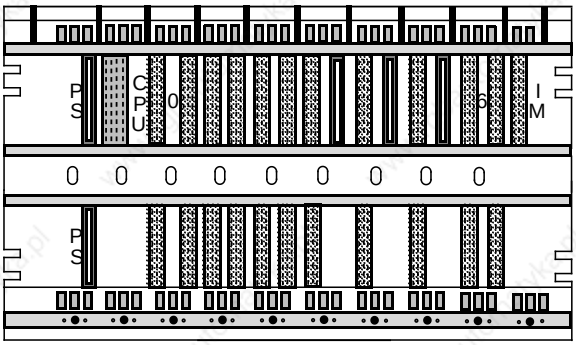
## Mounting Rack CR 700-2 for Central Controller 2 (6ES5 700-2LA12)




| Technical Specifications | | |
|---|---|---|
| Number of input/output modules that can be plugged in | maximum | 7 |
| Number of expansion units that can be connected - central | maximum | 3 |
| - distributed * | maximum | 63 |
| Dimensions w x h x d (mm) | | 483 x 303 x 47 (18.84 in. x 11.82 in. x 1.83 in.) |
| Weight | | 5 kg (11.02 lb.) |

* see Section 3.2.6

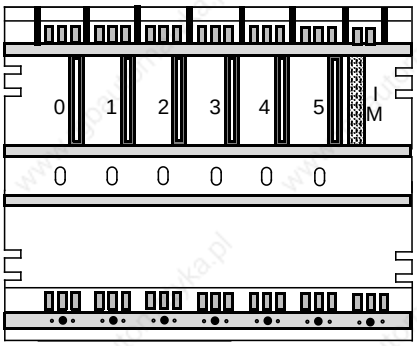
## Mounting Rack CR 700-3 for Central Controller 3 (6ES5 700-3LA12)




| Technical Specifications | | |
|---|---|---|
| Number of input/output modules that can be plugged in | maximum | 11 |
| Number of expansion units that can be connected | | |
| - central connection | maximum | 3 |
| - distributed * | maximum | 63 |
| Dimensions w x h x d (mm) | | 483 x 303 x 47 (18.84 in. x 11.82 in. x 1.83 in.) |
| Weight | | 5 kg (11.02 lb.) |

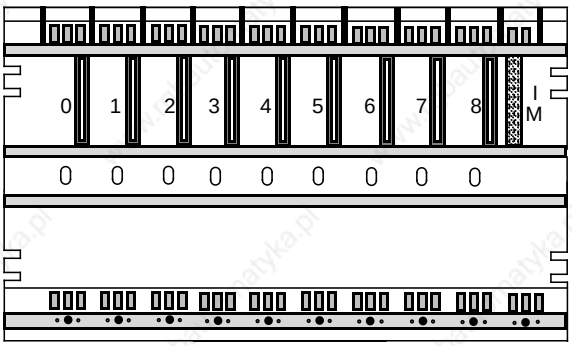
* see Section 3.2.6

## Mounting Rack ER 701-0 for Expansion Unit 0 (6ES5 701-0LA11)



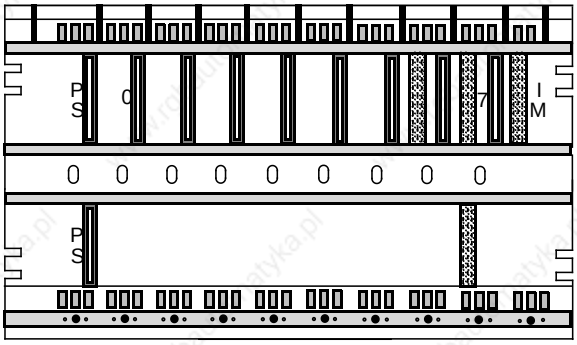

| Technical Specifications | | |
|---|---|---|
| Number of input/output modules that can be plugged in | maximum | 6 |
| Interface module - central connection | | IM 305/IM 306 |
| Interrupt evaluation | | not possible |
| Dimensions w x h x d (mm) | | 353 x 303 x 47 |
| Weight | | 4 kg (8.82 lb.) |

## Mounting Rack ER 701-1 for Expansion Unit 1 (6ES5 701-1LA12)



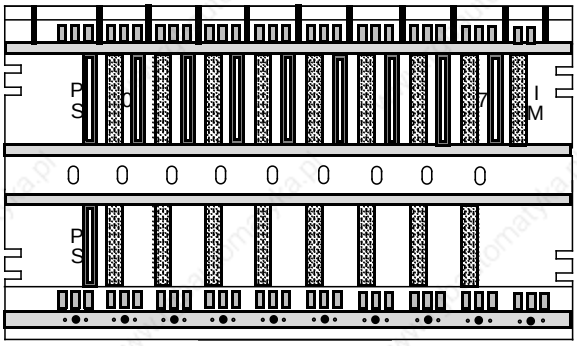

| Technical Specifications | | |
|---|---|---|
| Number of input/output modules that can be plugged in | maximum | 9 |
| Interface module | | |
| - central connection | | IM 305/IM 306 |
| Interrupt evaluation | | not possible |
| Dimensions w x h x d (mm) | | 483 x 303 x 47 (18.84 in. x 11.82 in. x 1.83 in.) |
| Weight | | 5 kg (11.02 lb.) |

## Mounting Rack ER 701-2 for Expansion Unit 2 (6ES5 701-2LA12)




| Technical Specifications | | |
|---|---|---|
| Number of input/output modules that can be plugged in | maximum | 8 |
| Interface module | | |
| - central connection | | IM 306 |
| - distributed connection | | AS 310/AS 311 IM 314/IM 317/ IM 318 |
| Interrupt evaluation | | not possible |
| Dimensions w x h x d (mm) | | 483 x 303 x 47 (18.84 in. x 11.82 in. x 1.83 in.) |
| Weight | | 5 kg (11.02 lb.) |

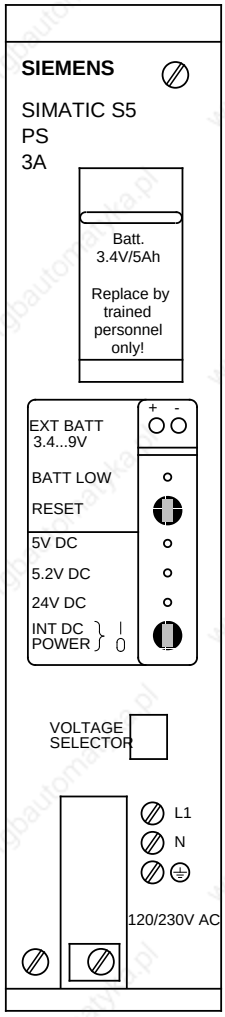
## Mounting Rack ER 701-3 for Expansion Unit 3 (6ES5 701-3LA12)

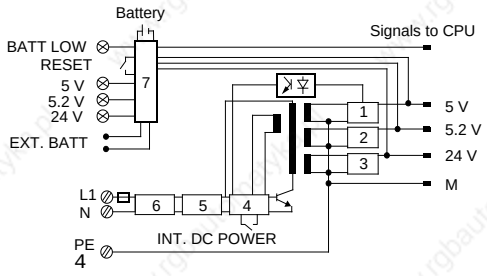



| Technical Specifications | | |
|---|---|---|
| Number of input/output modules that can be that can be plugged in | maximum | 8 |
| Interface module | | |
| - central connection | | IM 306 |
| - distributed connection | | AS 310/AS 311 IM 314/IM 317/ IM 318 |
| Interrupt evaluation | | possible with IM 307/317 |
| Dimensions w x h x d (mm) | | 483 x 303 x 47 (18.84 in. x 11.82 in. x 1.83 in.) |
| Weight | | 5 kg (11.02 lb.) |

## 15.2.2 Power Supply Modules

### Power Supply Module PS 951 120/230 V AC; 5 V, 3 A (6ES5 951-7LB21)



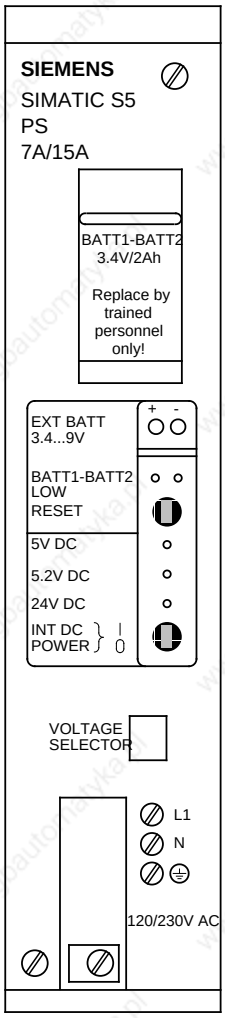



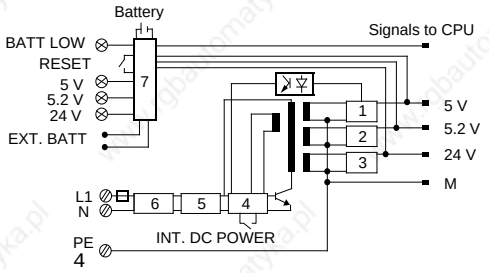

1 Chopper controller
2 Linear controller
3 Linear controller
4 Control electronics
5 Rectifier
6 Radio interference suppression filter
7 Monitoring electronics

**Block diagram**

| **Technical specifications** | | |
|---|---|---|
| **Input voltage L1** | | |
| **- Rated value** | | **120/230 V AC** |
| **- Permissible range** | | **94 to 132 V/ 187 to 264 V** |
| **Line frequency** | | |
| **- Rated value** | | **50 Hz** |
| **- Permissible range** | | **47 to 63 Hz** |
| **Input current at 120/230 V** | | |
| **- Rated value** | | **0.55/0.33 A** |
| **- Inrush current** | **max.** | **15×$I_N$** |
| **- $I^2t$** | | **0.135 $A^2s$** |
| **Power consumption (active power)** | | **44.8 W** |
| **Output voltage** | | |
| **- Rated value** | | **5 V** |
| **- Tolerance** | | **±1.5%** |
| **Output current** | | |
| **- Rated value without fan** | | **3 A** |
| **- Rated value with fan** | | **3 A** |
| **- Permissible range** | | **0.3 A to 3 A** |
| **Output voltage (PG/OP)** | | |
| **- Rated value** | | **5.2 V** |
| **- Tolerance** | | **±1.5%** |
| **Output current** | | **max. 1 A** |
| **Output voltage (auxiliary voltage)** | | |
| **- Rated value** | | **24 V** |
| **- Tolerance** | | **±5%** |
| **Output current** | | **max. 0.2 A** |
| **Backup battery** | | **Lithium battery, size C (3.6 V/5 Ah)** |
| **- Backup time** | **min.** | **1 year (at 0.3 mA, 25°C and uninterrupted backup)** |
| **Mains buffering (at $L1_{min}$)** | **min.** | **20 ms** |
| **Short-circuit protection** | | **Electronic** |
| **Fault indicator** | | **No** |
| **Fuse (primary circuit)** | | **Integral** |
| **Class of protection** | | **Class 1** |
| **Galvanic isolation** | | **Yes** |
| **Isolation rating** | | **Safe electrical isolation according to VDE 0160** |
| **- Tested with** | | **2700 V DC** |
| **Initial discharge current to VDE 0160 at 230 V AC** | | **2.6 mA** |
| **RI specifications** | | **A according to VDE 0871** |
| **Power losses of the module** | **typ.** | **19.8 W** |
| **Weight** | **approx.** | **1.6 kg (3.53 lb.)** |

## Power Supply Module PS 951 120/230 V AC; 5 V, 7/15 A (6ES5 951-7LD21)



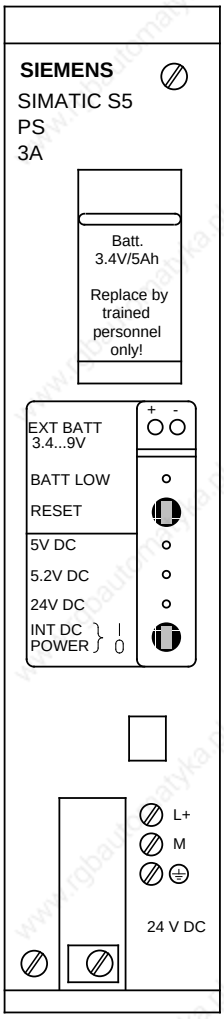

1 Chopper controller
2 Linear controller
3 Linear controller
4 Control electronics
5 Rectifier
6 Radio interference suppression filter
7 Monitoring electronics

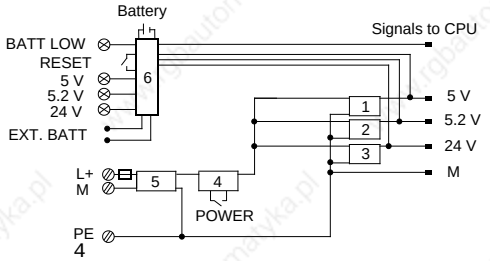
**Block diagram**

**Technical specifications**

| | | |
|---|---|---|
| **Input voltage L1** | | |
| **- Rated value** | | **120/230 V AC** |
| **- Permissible range** | | **94 to 132 V**<br>**187 to 264 V** |
| **Line frequency** | | |
| **- Rated value** | | **50 Hz** |
| **- Permissible range** | | **47 to 63 Hz** |
| **Input current at 120/230 V** | | |
| **- Rated value** | | **1.4/0.8 A** |
| **- Inrush current** | **max.** | **15×$I_N$** |
| **- $I^2t$** | | **1.8 $A^2s$** |
| **Power consumption (active power)** | | **133 W** |
| **Output voltage** | | |
| **- Rated value** | | **5 V** |
| **- Tolerance** | | **±1.5%** |
| **Output current** | | |
| **- Rated value without fan** | | **7 A** |
| **- Rated value with fan** | | **15 A** |
| **- Permissible range** | | **0.3 to 15 A** |
| **Output voltage (PG/OP)** | | |
| **- Rated value** | | **5.2 V** |
| **- Tolerance** | | **±1.5%** |
| **Output current** | | **max. 2.5 A** |
| **Backup battery** | | **2 lithium batteries, size AA (3.6 V/2x1.75 Ah)** |
| **- Backup time** | **min.** | **1 year (at 0.3 mA, 25°C and uninterrupted backup)** |
| **Mains buffering (at $L1_{min}$)** | **min.** | **20 ms** |
| **Output voltage (auxiliary voltage)** | | |
| **- Rated value** | | **24 V** |
| **- Tolerance** | | **±5%** |
| **Output current** | | **max. 0.35 A** |
| **Short-circuit protection** | | **Electronic** |
| **Fault indicator** | | **No** |
| **Fuse (primary circuit)** | | **Integral** |
| **Class of protection** | | **Class 1** |
| **Galvanic isolation** | | **Yes** |
| **Isolation rating** | | **Safe electrical isolation according to VDE 0160** |
| **- Tested with** | | **2700 V DC** |
| **Initial discharge current to VDE 0160 at 230 V AC** | | **2.6 mA** |
| **RI specifications** | | **A according to VDE 0871** |
| **Power losses of the module** | **typ.** | **36 W** |
| **Weight** | **approx.** | **1.9 kg (4.19 lb.)** |

## Power Supply Module PS 951 24 V DC; 5 V, 3 A (6ES5 951-7NB21)



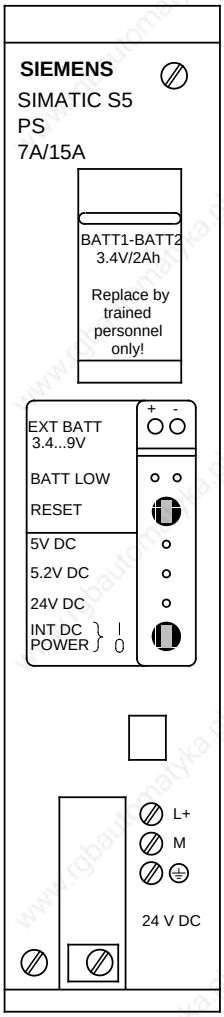

1 Chopper controller
2 Chopper controller
3 Linear controller
4 Control electronics
5 Radio interference suppression filter
6 Monitoring electronics

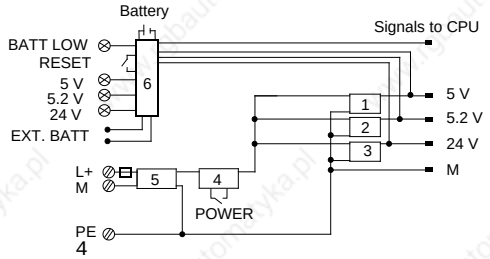
**Block diagram**

**Technical specifications**

| | | |
|---|---|---|
| **Input voltage L+** | | |
| **- Rated value** | | **24 V DC** |
| **- Permissible range** | | **19.2 to 30 V** |
| **Input current at 24 V** | | |
| **- Rated value** | | **1.51 A** |
| **- Inrush current** | **max.** | **15×$I_N$** |
| **- $I^2t$** | | **0.4 $A^2s$** |
| **Power consumption** | | **36.2 W** |
| **Output voltage** | | |
| **- Rated value** | | **5 V** |
| **- Tolerance** | | **±1.5%** |
| **Output current** | | |
| **- Rated value without fan** | | **3 A** |
| **- Rated value with fan** | | **3 A** |
| **- Permissible range** | | **0.3 to 3 A** |
| **Output voltage (PG/OP)** | | |
| **- Rated value** | | **5.2 V** |
| **- Tolerance** | | **±1.5%** |
| **Output current** | | **max. 1 A** |
| **Backup battery** | | **Lithium battery, size C (3.6 V/5 Ah)** |
| **- Backup time** | **min.** | **1 year (at 0.3 mA, 25°C and uninterrupted backup)** |
| **Mains buffering (at $L+_{min}$)** | **min.** | **20 ms** |
| **Output voltage (auxiliary voltage)** | | |
| **- Rated value** | | **24 V** |
| **- Tolerance** | | **±5%** |
| **Output current** | | **max. 0.2 A** |
| **Short-circuit protection** | | **Electronic** |
| **Fault indicator** | | **No** |
| **Fuse (primary circuit)** | | **Integral** |
| **Class of protection** | | **Class 1** |
| **Galvanic isolation** | | **No** |
| **RI specifications** | | **A according to VDE 0871** |
| **Power losses of the module** | **typ.** | **11.2 W** |
| **Weight** | **approx.** | **1.6 kg (3.53 lb.)** |

## Power Supply Module PS 951 24 V DC; 5 V, 7/15 A (6ES5 951-7ND51)



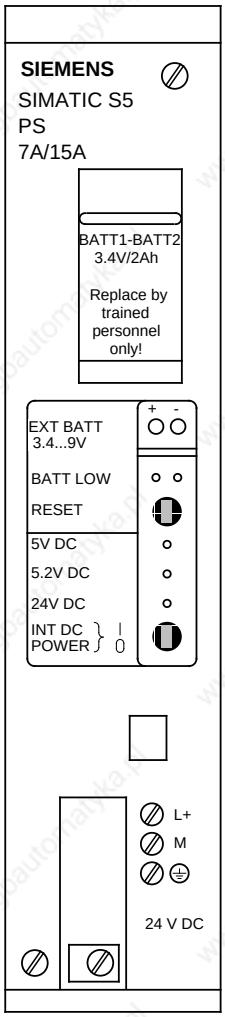

1 Chopper controller
2 Chopper controller
3 Linear controller
4 Control electronics
5 Radio interference suppression filter
6 Monitoring electronics

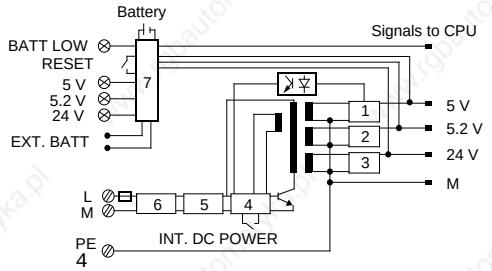
**Block diagram**

**Technical specifications**

| | | |
|---|---|---|
| **Input voltage L+** | | |
| **- Rated value** | | **24 V DC** |
| **- Permissible range** | | **19.2 to 30 V** |
| **Input current at 24 V** | | |
| **- Rated value** | | **5.04 A** |
| **- Inrush current** | **max.** | **15×$I_N$** |
| **- $I^2t$** | | **16 $A^2s$** |
| **Power consumption** | | **120.5 W** |
| **Output voltage** | | |
| **- Rated value** | | **5 V** |
| **- Tolerance** | | **±1.5%** |
| **Output current** | | |
| **- Rated value without fan** | | **7 A** |
| **- Rated value with fan** | | **15 A** |
| **- Permissible range** | | **0.3 to 15 A** |
| **Output voltage (PG/OP)** | | |
| **- Rated value** | | **5.2 V** |
| **- Tolerance** | | **±1.5%** |
| **Output current** | | **max. 2.5 A** |
| **Backup battery** | | **2 lithium batteries, size AA (3.6 V/ 2×1.75Ah)** |
| **- Backup time** | **min.** | **1 year (at 0.3 mA, 25°C and uninterrupted backup)** |
| **Mains buffering (at $L+_{min}$)** | **min.** | **20 ms** |
| **Output voltage (auxiliary voltage)** | | |
| **- Rated value** | | **24 V** |
| **- Tolerance** | | **±5%** |
| **Output current** | | **max. 0.35 A** |
| **Short-circuit protection** | | **Electronic** |
| **Fault indicator** | | **No** |
| **Fuse (primary circuit)** | | **Integral** |
| **Class of protection** | | **Class 1** |
| **Galvanic isolation** | | **No** |
| **RI specifications according to VDE** | | **A according to VDE 0871** |
| **Power losses of the module** | **typ.** | **24.1 W** |
| **Weight** | **approx.** | **1.7 kg (3.75 lb.)** |

## Power Supply Module PS 951 24 V DC; 5 V, 7/15 A (6ES5 951-7ND41)




1 Chopper controller
2 Linear controller
3 Linear controller
4 Control electronics
5 Rectifier
6 Radio interference suppression filter
7 Monitoring electronics

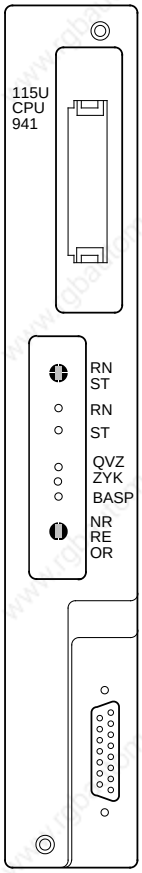
**Block diagram**

**Technical specifications**

| | | |
|---|---|---|
| **Input voltage L+** | | |
| **- Rated value** | | **24 V DC** |
| **- Permissible range** | | **19.2 to 30 V** |
| **Input current at 24 V** | | |
| **- Rated value** | | **5.6 A** |
| **- Inrush current** | **max.** | **15×$I_N$** |
| **- $I^2t$** | | **4.5 $A^2$ s** |
| **Power consumption** | | **134.4 W** |
| **Output voltage** | | |
| **- Rated value** | | **5 V** |
| **- Tolerance** | | **±1.5%** |
| **Output current** | | |
| **- Rated value without fan** | | **7 A** |
| **- Rated value with fan** | | **15 A** |
| **- Permissible range** | | **0.3 to 15 A** |
| **Output voltage (PG/OP)** | | |
| **- Rated value** | | **5.2 V** |
| **- Tolerance** | | **±1.5%** |
| **Output current** | | **max. 2.5 A** |
| **Backup battery** | | **2 lithium batteries, size AA (3.6 V/2 x 1.75 Ah)** |
| **- Backup time** | **min.** | **1 year (at 0.3 mA, 25°C and uninterrupted backup)** |
| **Mains buffering (at $L+_{min}$)** | **min.** | **20 ms** |
| **Output voltage (auxiliary voltage)** | | |
| **- Rated value** | | **24 V** |
| **- Tolerance** | | **±5%** |
| **Output current** | | **max. 0.35 A** |
| **Short-circuit protection** | | **Electronic** |
| **Fault indicator** | | **No** |
| **Fuse (primary circuit)** | | **Integral** |
| **Class of protection** | | **Class 1** |
| **Galvanic** | | **Yes** |
| **Isolation rating** | | **Safe electrical isolation according to VDE 0160** |
| **- Tested with** | | **2700 V DC** |
| **RI specifications** | | **A according to VDE 0871** |
| **Power losses of the module** | **typ.** | **38 W** |
| **Weight** | **approx.** | **1.7 kg (3.75 lb.)** |

## 15.2.3 Central Processing Units

**Central Processing Unit CPU 941** **(6ES5 941-7UB11)**

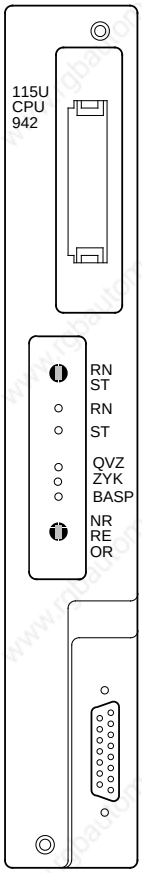



**Technical Specifications**

| | | |
|---|---|---|
| **Memory capacity (total)** | **maximum** | **9216 statements[1]** |
| **- internal memory** | **maximum** | **1024 statements[1]** |
| **- memory submodule (RAM)** | **maximum** | **8192 statements[1]** |
| **- memory submodule (EPROM)** | **maximum** | **8192 statements[1]** |
| **- memory submodule (EEPROM)** | **maximum** | **8192 statements[1]** |
| **Execution time** | | |
| **- per binary operation** | **approx.** | **2.2 µsec.** |
| **- per word operation** | **approx.** | **60 to 200 µsec.** |
| **Scan time monitoring** | **approx.** | **500 msec. (can be modified)** |
| **Flags** | | **2048; optionally half or all retentive [2]** |
| **Timers** | | |
| **- number** | | **128; optionally half or all retentive [2]** |
| **- range** | | **0.01 to 9990 sec.** |
| **Counters** | | |
| **- number** | | **128; optionally half or all retentive [2]** |
| **- range** | | **0 to 999 (up, down)** |
| **Digital inputs** | | |
| **Digital outputs - total** | **maximum** | **2048** |
| **Analog inputs** | | |
| **Analog outputs - total** | **maximum** | **128** |
| **Organization blocks** | **maximum** | **256** |
| **Program blocks** | **maximum** | **256** |
| **Function blocks** | **maximum** | **256 (can be assigned parameters)** |
| **Sequence blocks** | **maximum** | **256** |
| **Data blocks** | **maximum** | **254** |
| **Operations set** | **approx.** | **170 operations** |
| **Required backup current from the backup battery at power off** | | |
| **- internal RAM** | **approx.** | **100 µA** |
| **- RAM submodule** | **approx.** | **200 µA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **0.16 A** |
| **- from 24 V** | | |
| **(without programmer)** | | **0.04 A** |
| **(with programmer)** | | **0.06 A** |
| **Power losses of the module** | **typically** | **2 W** |
| **- with programmer** | **typically** | **2.5 W** |
| **Weight** | **approx.** | **0.8 kg (1.76 lb.)** |

**[1] A statement usually takes up two bytes in the program memory.**
**[2] Use back-up battery for retentive feature.**

## Central Processing Unit CPU 942 (6ES5 942-7UB11)

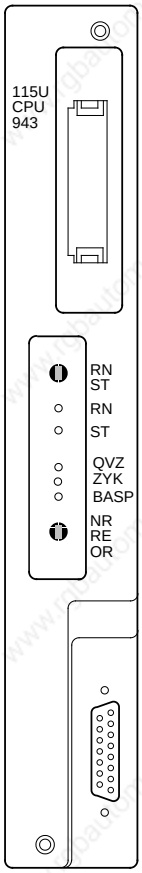



**Technical Specifications**

| | | |
|---|---|---|
| **Memory capacity (total)** | **maximum** | **21504 statements [1]** |
| **- internal memory** | **maximum** | **5120 statements [1]** |
| **- memory submodule (RAM)** | **maximum** | **16384 statements [1]** |
| **- memory submodule (EPROM)** | **maximum** | **16384 statements [1]** |
| **- memory submodule (EEPROM)** | **maximum** | **8192 statements [1]** |
| **Execution time** | | |
| **- per binary operation** | **approx.** | **1.6 µsec.** |
| **- per word operation** | **approx.** | **1.6 to 200 µsec.** |
| **Scan time monitoring** | **approx.** | **500 msec. (can be modified)** |
| **Flags** | | **2048; optionally half or all retentive [2]** |
| **Timers** | | |
| **- number** | | **128; optionally half or all retentive [2]** |
| **- range** | | **0.01 to 9990 sec.** |
| **Counters** | | |
| **- number** | | **128; optionally half or all retentive [2]** |
| **- range** | | **0 to 999 (up, down)** |
| **Digital inputs** | | |
| **Digital outputs - total** | **maximum** | **2048** |
| **Analog inputs** | | |
| **Analog outputs - total** | **maximum** | **128** |
| **Organization blocks** | **maximum** | **256** |
| **Program blocks** | **maximum** | **256** |
| **Function blocks** | **maximum** | **256 (can be assigned parameters)** |
| **Sequence blocks** | **maximum** | **256** |
| **Data blocks** | **maximum** | **254** |
| **Operations set** | **approx.** | **170 operations** |
| **Required backup current from the backup battery at power off** | | |
| **- internal RAM** | **approx.** | **100 µA** |
| **- RAM submodule** | **approx.** | **200 µA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **0.16 A** |
| **- from 24 V** | | |
| **(without programmer)** | | **0.04 A** |
| **(with programmer)** | | **0.06 A** |
| **Power losses of the module** | **typically** | **2 W** |
| **- with programmer** | **typically** | **2.5 W** |
| **Weight** | **approx.** | **0.8 kg (1.76 lb.)** |

**[1] A statement usually takes up two bytes in the program memory.**
**[2] Use back-up battery for retentive feature.**

## Central Processing Unit CPU 943 (with One Serial Interface) (6ES5 943-7UB11)

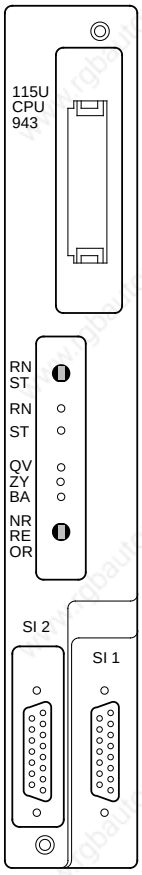



**Technical Specifications**

| | | |
|---|---|---|
| **Memory capacity (total)** | **maximum** | **24576 statements [1]** |
| **- internal memory** | **maximum** | **24576 statements [1]** |
| **- memory submodule (EPROM)** | **maximum** | **24576 statements [1]** |
| **- memory submodule (EEPROM)** | **maximum** | **8192 statements [1]** |
| **Execution time** | | |
| **- per binary operation** | **approx.** | **0.8 µsec.** |
| **- per word operation** | **approx.** | **0.8 to 160 µsec.** |
| **Scan time monitoring** | **approx.** | **500 msec. (can be modified)** |
| **Flags** | | **2048; optionally half or all retentive[2]** |
| **Timers** | | |
| **- number** | | **128; optionally half or all retentive [2]** |
| **- range** | | **0.01 to 9990 sec.** |
| **Counters** | | |
| **- number** | | **128; optionally half or all retentive [2]** |
| **- range** | | **0 to 999 (up, down)** |
| **Digital inputs** | | |
| **Digital outputs - total** | **maximum** | **2048** |
| **Analog inputs** | | |
| **Analog outputs - total** | **maximum** | **128** |
| **Organization blocks** | **maximum** | **256** |
| **Program blocks** | **maximum** | **256** |
| **Function blocks** | **maximum** | **256 (can be assigned parameters)** |
| **Sequence blocks** | **maximum** | **256** |
| **Data blocks** | **maximum** | **254** |
| **Operations set** | **approx.** | **170 operations** |
| **Required backup current from the backup battery at power off** | | |
| **- internal RAM** | **approx.** | **100 µA** |
| **- RAM submodule** | **approx.** | **200 µA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | **typically** | **0.2 A** |
| **- from 24 V** | | |
| **(without programmer)** | | **0.04 A** |
| **(with programmer)** | | **0.06 A** |
| **Power losses of the module** | **typically** | **2 W** |
| **- with PG** | **typically** | **2.5 W** |
| **Weight** | **approx.** | **0.8 kg (1.76 lb.)** |

**[1] A statement usually takes up two bytes in the program memory.**
**[2] Use back-up battery for retentive feature.**

## Central Processing Unit CPU 943 (with Two Serial Interfaces) (6ES5 943-7UB21)

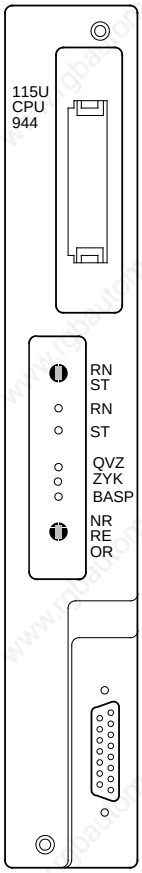



**Technical Specifications**

| | | |
|---|---|---|
| Memory capacity (total) | maximum | 24576 statements [1] |
| - internal memory | maximum | 24576 statements [1] |
| - memory submodule (EPROM) | maximum | 24576 statements [1] |
| - memory submodule (EEPROM) | maximum | 8192 statements [1] |
| Scan time monitoring | | |
| - per binary operation | approx. | 0.8 µsec. |
| - per word operation | approx. | 0.8 to 160 µsec. |
| Clock | | |
| - Accuracy $t_g$ | | ± 2sec. /day |
| - Temperature dependency $t_A$ (ambient termperature $T_U$ in °C) | | -3.5 x $(T_U$-15)$^2$ msec. /day |
| - e.g. tolerance at 40°C | | ± 2 sec.-3,5 x (40 - 15)$^2$ msec./day |
| | approx. | 0 to - 4 sec./day |
| Scan time monitoring | approx. | 500 msec. (can be modified) |
| Flags | | 2048; optionally half or all retentive [2] |
| Timers | | |
| - number | | 128; optionally half or all retentive [2] |
| - range | | 0,01 to 9990 sec. |
| Counters | | |
| - number | | 128; optionally half or all retentive [2] |
| - range | | 0 to 999 (up, down) |
| Digital inputs | | |
| Digital outputs - total | maximum | 2048 |
| Analog inputs | | |
| Analog outputs - total | maximum | 128 |
| Organization blocks | maximum | 256 |
| Program blocks | maximum | 256 |
| Function blocks | maximum | 256 (can be assigned parameters) |
| Sequence blocks | maximum | 256 |
| Data blocks | maximum | 254 |
| Operations set | approx. | 170 operations |
| Required backup current from the backup battery at power off | | |
| - internal RAM | approx. | 100 µA |
| - RAM submodule | approx. | 200 µA |
| Current consumption | | |
| - from 5 V (internal) | typically | 0.45 A |
| - from 24 V | | |
| (without programmer) | | 0.08 A |
| (with programmer) | | 0.12 A |
| Power losses of the module | typically | 4.5 W |
| - with two programmers | typically | 5.5 W |
| Weight | approx. | 0.8 kg (1.76 lb.) |

[1] A statement usually takes up two bytes in the program memory
[2] In the case of battery backup

## Central Processing Unit CPU 944 (with One Serial Interface) (6ES5 944-7UB11)

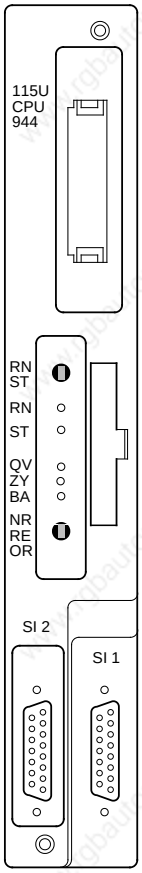



**Technical Specifications**

| | | |
|---|---|---|
| **Memory capacity (total)** | **maximum** | **49152 statements [1]** |
| **- internal memory** | **maximum** | **49152 statements [1]** |
| **- memory submodule (EPROM)** | **maximum** | **49152 statements [1]** |
| **- memory submodule (EEPROM)** | **maximum** | **8192 statements [1]** |
| **Execution time** | | |
| **- per binary operation** | **approx.** | **0.8 µsec.** |
| **- per word operation** | **approx.** | **0.8 to 3.6 µsec.** |
| **Scan time monitoring** | **approx.** | **500 msec. (can be modified)** |
| **Flags** | | **2048; optionally half or all retentive [2]** |
| **Timers** | | |
| **- number** | | **128; optionally half or all retentive [2]** |
| **- range** | | **0.01 to 9990 sec.** |
| **Counters** | | |
| **- number** | | **128; optionally half or all retentive [2]** |
| **- range** | | **0 to 999 (up, down)** |
| **Digital inputs** | | |
| **Digital outputs - total** | **maximum** | **2048** |
| **Analog inputs** | | |
| **Analog outputs - total** | **maximum** | **128** |
| **Organization blocks** | **maximum** | **256** |
| **Program blocks** | **maximum** | **256** |
| **Function blocks** | **maximum** | **256 (can be assigned parameters)** |
| **Sequence blocks** | **maximum** | **256** |
| **Data blocks** | **maximum** | **254** |
| **Operations set** | **approx.** | **170 operations** |
| **Required backup current from the backup battery at power off** | | |
| **- internal RAM** | **approx.** | **100 µA** |
| **- RAM submodule** | **approx.** | **200 µA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | **typically** | **0.2 A** |
| **- from 24 V** | | |
| **(without programmer)** | | **0.04 A** |
| **(with programmer)** | | **0.06 A** |
| **Power losses of the module** | **typically** | **2 W** |
| **- with programmer** | **typically** | **2.5 W** |
| **Weight** | **approx.** | **0.8 kg (1.76 lb.)** |

**[1] A statement usually takes up two bytes in the program memory**
**[2] In the case of battery backup**

## Central Processing Unit CPU 944 (with Two Serial Interfaces) (6ES5 944-7UB21)



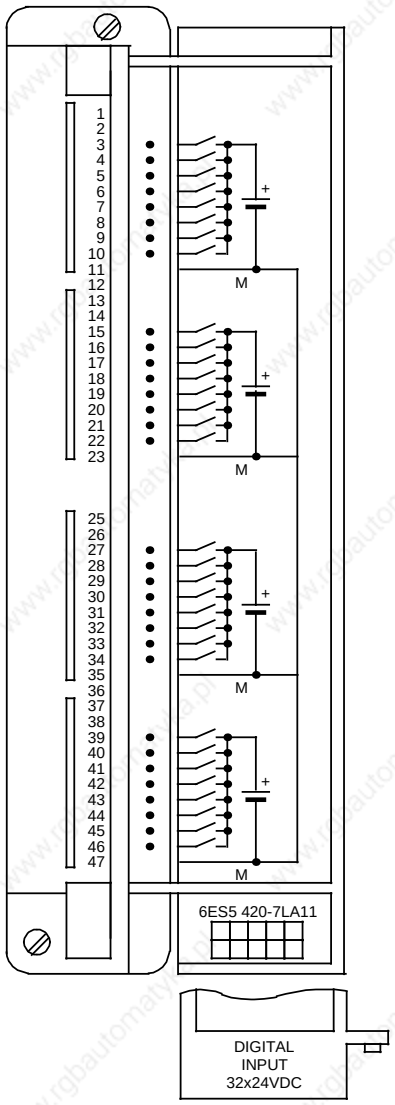

**Technical Specifications**

| | | |
|---|---|---|
| **Memory capacity (total)** | **maximum** | **49152 statements [1]** |
| **- internal memory** | **maximum** | **49152 statements [1]** |
| **- memory submodule (EPROM)** | **maximum** | **49152 statements [1]** |
| **- memory submodule (EEPROM)** | **maximum** | **8192 statements [1]** |
| **Execution time** | | |
| **- per binary operation** | **approx.** | **0.8 µsec.** |
| **- per word operation** | **approx.** | **0.8 to 3.6 µsec.** |
| **Clock** | | |
| **- Accuracy $t_g$** | | **±2 sec. /day** |
| **- Temperature dependency $t_A$ (ambient temperature $T_U$ in °C)** | | **- 3.5 x $(T_U-15)^2$ msec./day** |
| **- e.g. tolerance at 40 °C** | | **±2 sec. - 3.5 x $(40 - 15)^2$ msec /day** |
| | **approx.** | **0 to - 4 sec./day** |
| **Scan time monitoring** | **approx.** | **500 msec. (can be modified)** |
| **Flags** | | **2048; optionally half or all retentive [2]** |
| **Timers** | | |
| **- number** | | **128; optionally half or all retentive [2]** |
| **- range** | | **0.01 to 9990 sec.** |
| **Counters** | | |
| **- number** | | **128; optionally half or all retentive [2]** |
| **- range** | | **0 to 999 (up, down)** |
| **Digital inputs** | | |
| **Digital outputs - total** | **maximum** | **2048** |
| **Analog inputs** | | |
| **Analog outputs - total** | **maximum** | **128** |
| **Organization blocks** | **maximum** | **256** |
| **Program blocks** | **maximum** | **256** |
| **Function blocks** | **maximum** | **256 (can be assigned parameters)** |
| **Sequence blocks** | **maximum** | **256** |
| **Data blocks** | **maximum** | **254** |
| **Operations set** | **approx.** | **170 operations** |
| **Required backup current from the backup battery at power off** | | |
| **- internal RAM** | **approx.** | **100 µA** |
| **- RAM submodule** | **approx.** | **200 µA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | **typically** | **0.45 A** |
| **- from 24 V** | | |
| **(without programmer)** | | **0.08 A** |
| **(with 2 programmers)** | | **0.12 A** |
| **Power losses of the module** | **typically** | **4.5 W** |
| **- with 2 programmers** | **typically** | **5.5 W** |
| **Weight** | **approx.** | **0.8 kg (1.76 lb.)** |

**[1] A statement usually takes up two bytes in the program memory**
**[2] Use back-up battery for retentive feature**

## 15.2.4 Digital Input Modules

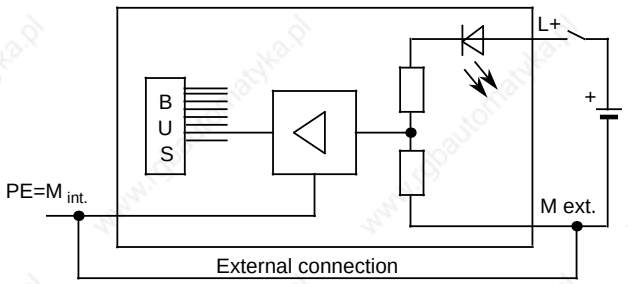
### Digital Input Module 32 x 24 V DC, Nonfloating (6ES5 420-7LA11)




**Technical Specifications**

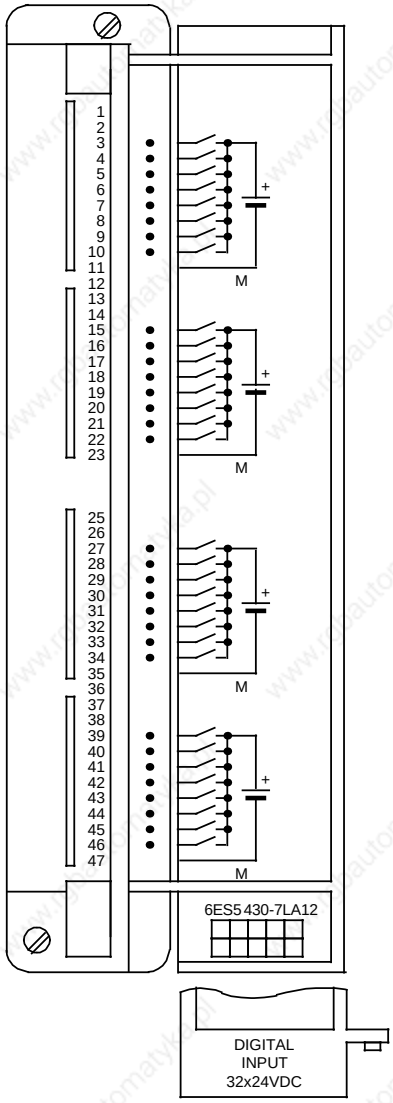
| | | |
|---|---|---|
| **Number of inputs** | | **32** |
| **Floating** | | **no** |
| **Input voltage L+** | | |
| **- rated value** | | **24 V DC** |
| **- for "0" signal** | | **- 30 to+5 V** |
| **- for "1" signal** | | **13 to 30 V** |
| **Input current** | | |
| **- for "1" signal** | **typically** | **8.5 mA** |
| **Response time** | | |
| **- from "0" to "1"** | | **1.4 to 5 msec.** |
| **- from "1" to "0"** | | **1.4 to 5 msec.** |
| **Cable length** | | |
| **- shielded** | **maximum** | **1000 m** |
| **- unshielded** | **maximum** | **600 m** |
| **Isolation rating** | | **according to VDE 0160** |
| **Connection of 2-wire BERO proximity switches** | | **possible** |
| **- leakage (quiescent) current** | | **1.5 mA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **5 mA** |
| **Power losses of the module** | **typically** | **6.5 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

**Terminal Assignment**

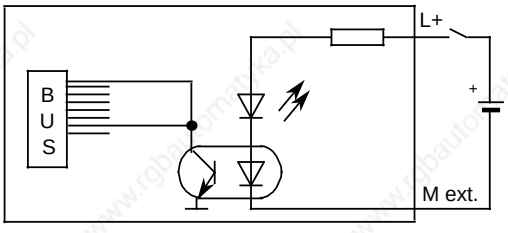



**Block diagram**

## Digital Input Module 32 x 24 V DC, Floating (6ES5 430-7LA12)




Terminal Assignment

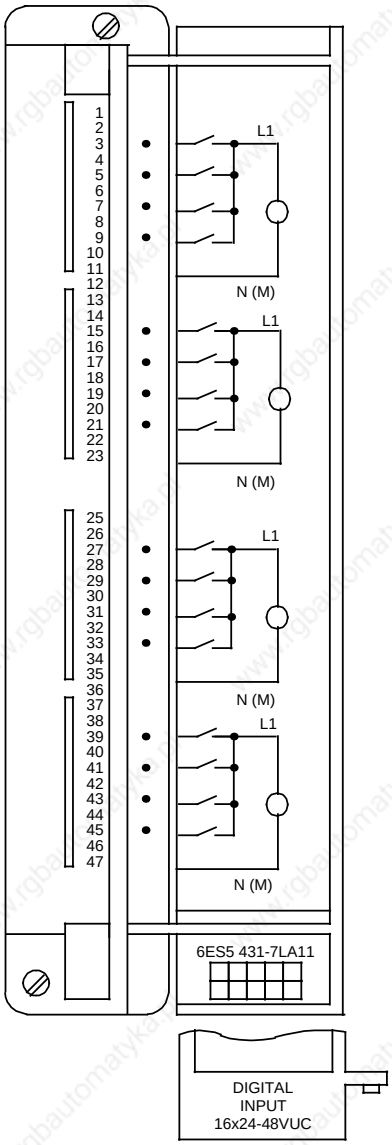



Block diagram

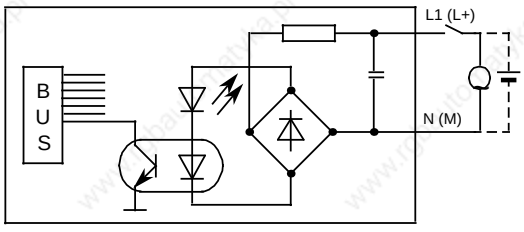
**Technical Specifications**

| | | |
|---|---|---|
| **Number of inputs** | | **32** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **8** |
| **Input voltage L+** | | |
| **- rated value** | | **24 V DC** |
| **- for "0" signal** | | **- 30 to+5V** |
| **- for "1" signal** | | **13 to 30V** |
| **Input current** | | |
| **- for "1" signal** | **typically** | **8.5 mA** |
| **Response time** | | |
| **- from "0" to "1"** | **typically** | **2.6 msec.;** |
| | **maximum** | **5.5 msec.** |
| **- from "1" to "0"** | **typically** | **2.3 msec.;** |
| | **maximum** | **5 msec.** |
| **Cable length** | | |
| **- shielded** | **maximum** | **1000 m (3281 ft.)** |
| **- unshielded** | **maximum** | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage** | | |
| **(between groups)** | | **30 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V** |
| **Rated isolation voltage** | | |
| **(L+ to ⏚ )** | | **30 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V** |
| **Connection of 2-wire** | | |
| **BERO proximity switches** | | **possible** |
| **- quiescent current** | | **1.5 mA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **5 mA** |
| **Power losses of** | | |
| **the module** | **typically** | **6.5 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

## Digital Input Module 16 x 24 to 48 V UC (6ES5 431-7LA11)




**Terminal Assignment**



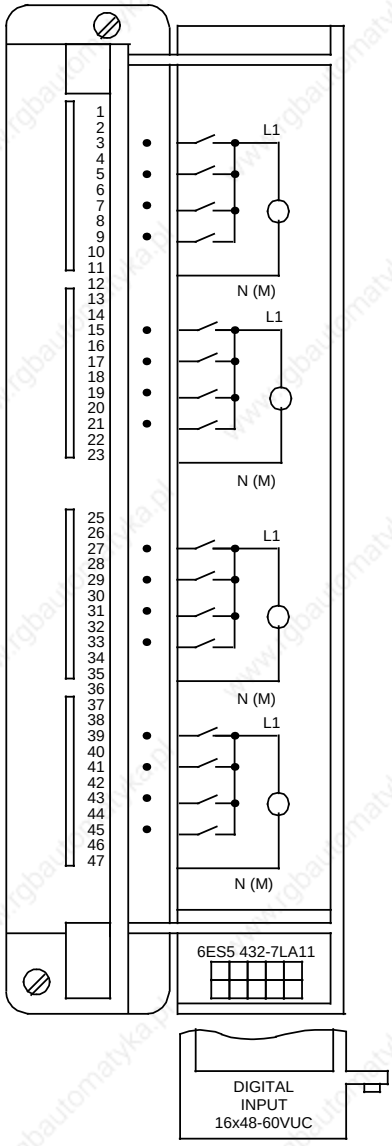

**Block diagram**

**Technical Specifications**

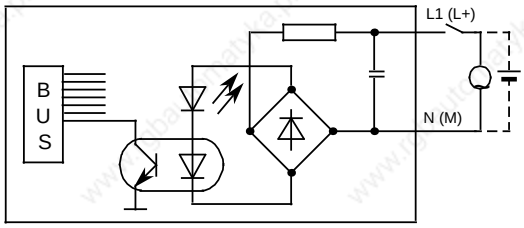
| | | |
|---|---|---|
| **Number of inputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **4** |
| **Input voltage L+** | | |
| **- rated value** | | **24 to 48 V UC** |
| **- frequency** | | **0 to 63 Hz** |
| **- for "0" signal** | | **0 to 5 V** |
| **- for "1" signal** | | **13 to 60 V** |
| **Input current at "1" signal** | | |
| **- for 24 V AC** | **typically** | **8.5 mA** |
| **- for 24 V DC** | **typically** | **9.0 mA** |
| **- for 48 V AC** | **typically** | **10.5 mA** |
| **- for 48 V DC** | **typically** | **10.5 mA** |
| **Response time** | | |
| **- from "0" to "1"** | | **2 to 13 msec.** |
| **- from "1" to "0"** | | **10 to 25 msec.** |
| **Cable length** | | |
| **- shielded** | **maximum** | **1000 m** |
| **- unshielded** | **maximum** | **600 m** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage [1]** | | |
| **(between groups)** | | **60V** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V** |
| **Rated isolation voltage** | | |
| **(L1 to ⏚ )** | | **250V** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V** |
| **Connection of 2-wire BERO proximity switches** | | **possible** |
| **- quiescent current** | | **2 mA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **5 mA** |
| **Power losses of the module** | **typically** | **9 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

**[1] Connection of different phases is not permissible.**

## Digital Input Module 16 x 48 to 60 V UC, Floating (6ES5 432-7LA11)




**Terminal Assignment**



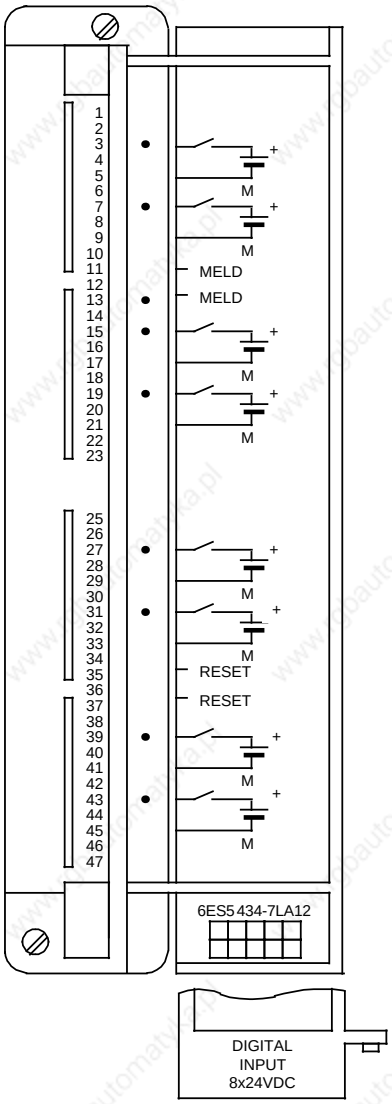

**Block diagram**

**Technical Specifications**

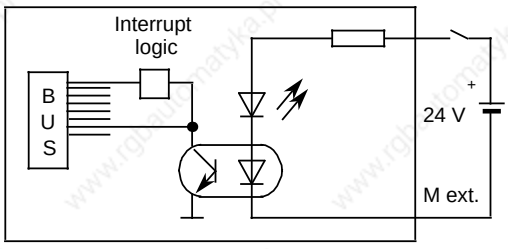
| | | |
|---|---|---|
| **Number of inputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **4** |
| **Input voltage L1** | | |
| **- rated value** | | **48 to 60 V UC** |
| **- frequency** | | **0 to 63 Hz** |
| **- for "0" signal** | | **0 to 10 V** |
| **- for "1" signal** | | **30 to 72 V** |
| **Input current at "1" signal** | | |
| **- for 48 V AC /50 Hz** | **typically** | **8.5 mA** |
| **- for 48 V DC** | **typically** | **9.5 mA** |
| **- for 60 V AC /50 Hz** | **typically** | **9.5 mA** |
| **- for 60 V DC** | **typically** | **10.0 mA** |
| **Response time** | | |
| **- from "0" to "1"** | | **2 to 13 msec.** |
| **- from "1" to "0"** | | **10 to 25 msec.** |
| **Cable length** | | |
| **- shielded** | **maximum** | **1000 m (3281 ft.)** |
| **- unshielded** | **maximum** | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage[1] (between groups)** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V** |
| **Rated isolation voltage (L1 to ⏚ )** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V** |
| **Connection of 2-wire BERO proximity switches** | | **possible** |
| **- quiescent current** | | **5 mA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **5 mA** |
| **Power losses of the module** | **typically** | **10 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

[1] **Connection of different phases is not permissible.**

## Digital Input Module 8 x 24 V DC (with P Interrupt), Floating (6ES5 434-7LA12)




**Terminal Assignment**

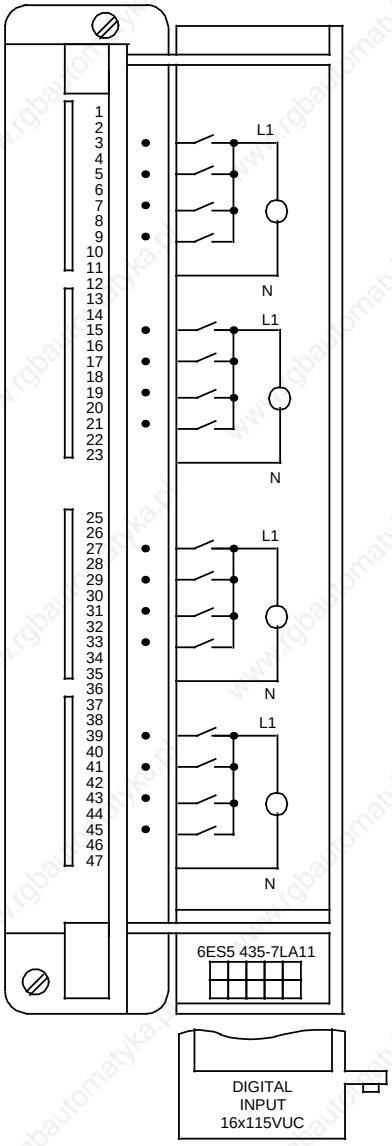



**Block diagram**

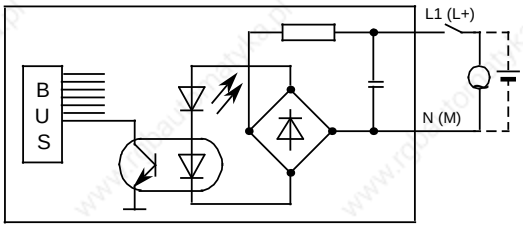
**Technical Specifications**

| | | |
|---|---|---|
| **Number of inputs** | | **8** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **1** |
| **Input voltage L+** | | |
| **- rated value** | | **24 V DC** |
| **- for "0" signal** | | **- 30 to +5 V** |
| **- for "1" signal** | | **13 to 30 V** |
| **Input current at "1" signal** | | |
| **- for 24 V DC** | **typically** | **8.5 mA** |
| **Response time** | | |
| **- from "0" to "1"** | | **0.5 to 1.5 msec.** |
| **- from "1" to "0"** | | **0.5 to 1.5 msec.** |
| **Interrupt signal (external)** | | **latching relay contact (permissible load: max. 0.2 A 50 V DC - switching capacity max. 20 W or 35 VA)** |
| **Interrupt signal (internal)** | | **via bus line PRAL-N** |
| **Acknowledgement** | | **external via input reset 24 V DC** |
| **Cable length** | | |
| **- shielded** | **maximum** | **1000 m (3281 ft.)** |
| **- unshielded** | **maximum** | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage (between groups)** | | **30V** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V** |
| **Rated isolation voltage (L+ to ⏚ )** | | **30 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V** |
| **Connection of 2-wire BERO proximity switches** | | **possible** |
| **- quiescent current** | **maximum** | **1.5 mA** |
| **Current Consumption** | | |
| **- from 5 V (internal)** | | **<70 mA** |
| **Power losses of the module** | **typically** | **2 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

## Digital Input Module 16 x 115 V UC, Floating (6ES5 435-7LA11)




**Terminal Assignment**



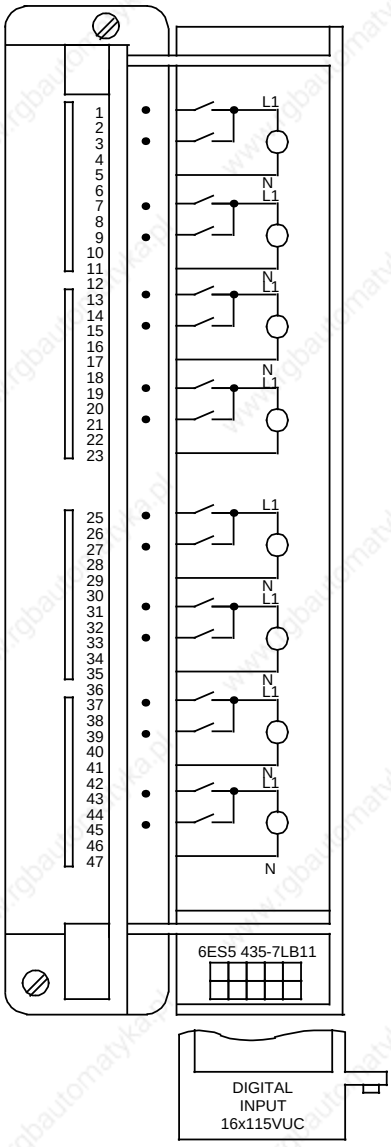

**Block diagram**

**Technical Specifications**

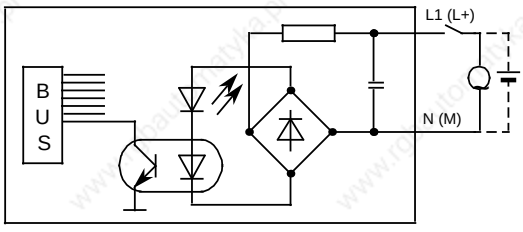
| | | |
|---|---|---|
| **Number of inputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **4** |
| **Input voltage L1** | | |
| **- rated value** | | **115 V AC** |
| **- frequency** | | **47 to 63 Hz** |
| **- for "0" signal** | | **0 to 40 V** |
| **- for "1" signal** | | **85 to 135 V** |
| **Input current at "1" signal** | | |
| **- for AC, 50 Hz** | **typically** | **15 mA** |
| **- for DC** | **typically** | **6 mA** |
| **Response time** | | |
| **- from "0" to "1"** | | **2 to 13 msec.** |
| **- from "1" to "0"** | | **10 to 25 msec.** |
| **Cable length** | | |
| **- shielded** | | **1000 m (3281 ft.)** |
| **- unshielded** | | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage [1] (between groups)** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V** |
| **Rated isolation voltage (L1 to ⏚ )** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V** |
| **Connection of 2-wire BERO proximity switches** | | **possible** |
| **- quiescent current** | | **5 mA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **5 mA** |
| **Simultaneity factor (per group, L1=135 V)** | | |
| **- at 25 °C** | | **100 %** |
| **- at 55 °C** | | **75 %** |
| **Power losses of the module** | **typically** | **11 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

[1] **Connection of different phases is permissible.**

## Digital Input Module 16 x 115 V UC (6ES5 435-7LB11)




**Terminal Assignment**



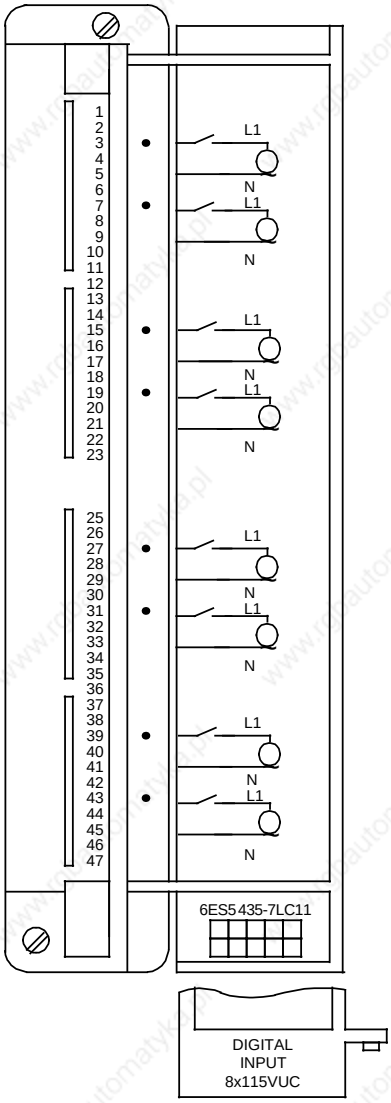

**Block diagram**

**Technical Specifications**

| | | |
|---|---|---|
| **Number of inputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in goups of** | | **2** |
| **Input voltage L1** | | |
| **- rated value** | | **115 V AC** |
| **- frequency** | | **47 to 63 Hz** |
| **- for "0" signal** | | **0 to 40 V** |
| **- for "1" signal** | | **85 to 135 V** |
| **Input current at "1" signal** | | |
| **- for AC, 50 Hz** | **typically** | **10 mA** |
| **- for DC** | **typically** | **6 mA** |
| **Response time** | | |
| **- from "0" to "1"** | | **2 to 13 msec.** |
| **- from "1" to "0"** | | **10 to 25 msec.** |
| **Cable length** | | |
| **- shielded** | | **1000 m (3281 ft.)** |
| **- unshielded** | | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage [1] (between groups)** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V** |
| **Rated isolation voltage (L1 to ⏚ )** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V** |
| **Connection of 2-wire BERO proximity switches** | | **possible** |
| **- quiescent current** | | **5mA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **5mA** |
| **Simultaneity factor (per group, L1=135 V)** | | |
| **- at 25 °C** | | **100 %** |
| **- at 55 °C** | | **75 %** |
| **Power losses of the module** | **typically** | **11 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

**[1] Connection of different phases is permissible.**

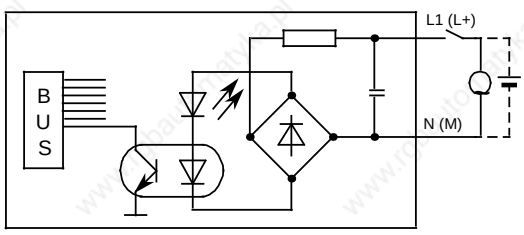
## Digital Input Module 8 x 115V UC (6ES5 435-7LC11)




**Technical Specifications**

| | | |
|---|---|---|
| **Number of inputs** | | **8** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **1** |
| **Input voltage L1** | | |
| **- rated value** | | **115 V UC** |
| **- frequency** | | **47 to 63 Hz** |
| **- for "0" signal** | | **0 to 40 V** |
| **- for "1" signal** | | **85 to 135 V** |
| **Input current at** | | |
| **"1" signal** | **AC typically** | **10 mA** |
| | **DC typically** | **6 mA** |
| **Response time** | | |
| **- from "0" to "1"** | | **2 to 13 msec.** |
| **- from "1" to "0"** | | **10 to 25 msec.** |
| **Cable length** | | |
| **- shielded** | | **1000 m (3281 ft.)** |
| **- unshielded** | | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage [1)]** | | |
| **(between groups)** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **2700 V** |
| **Rated isolation voltage** | | |
| **(L1 to ⏚ )** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **2700 V** |
| **Connection of 2-wire** | | |
| **BERO proximity switches** | | **possible** |
| **- quiescent current** | | **5 mA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **5 mA** |
| **Power losses of** | | |
| **the module** | **typically** | **5.5 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

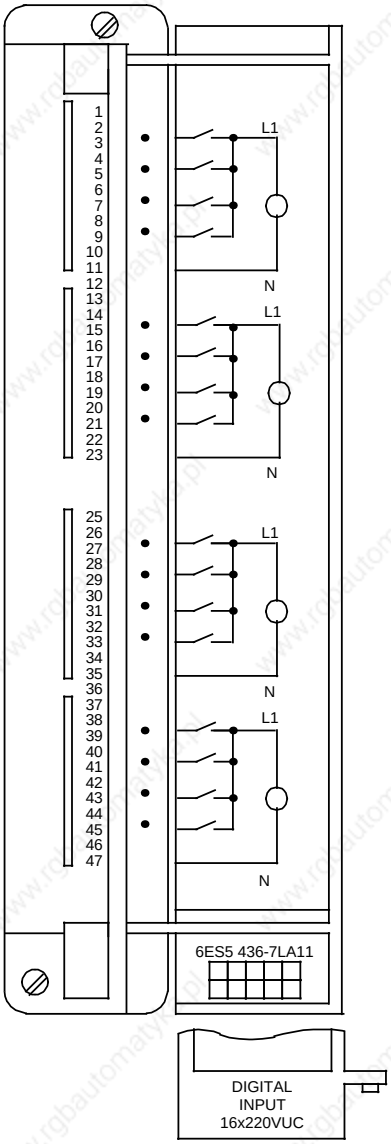
**Terminal Assignment**



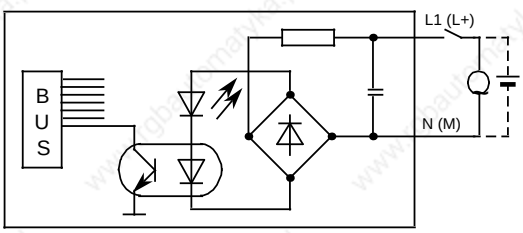

**Block diagram**

[1] **Connection of different phases is not permissible.**

## Digital Input Module 16 x 230 V UC, Floating (6ES5 436-7LA11)




**Terminal Assignment**



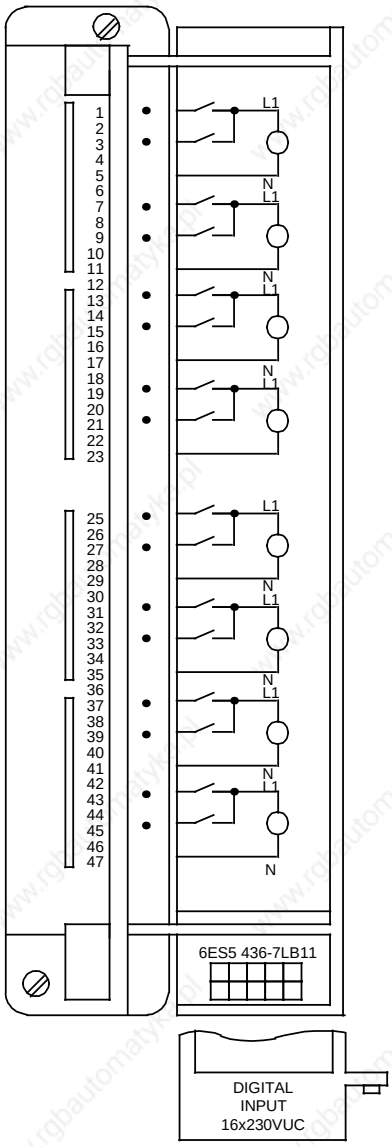

**Block diagram**

**Technical Specifications**

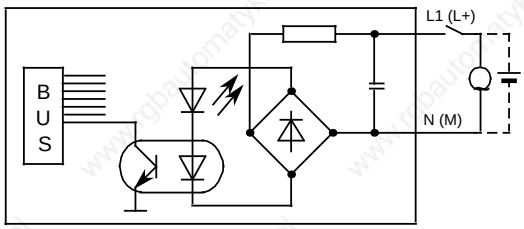
| | | |
|---|---|---|
| **Number of inputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **4** |
| **Input voltage L1** | | |
| **- rated value** | | **230 V AC** |
| **- frequency** | | **47 to 63 Hz** |
| **- for "0" signal** | | **0 to 70 V** |
| **- for "1" signal** | | **170 to 264 V** |
| **Input current** | | |
| **at "1" signal** | | |
| **- for AC, 50 Hz** | **typically** | **15 mA** |
| **- for DC** | **typically** | **2.2 mA** |
| **Response time** | | |
| **- from "0" to "1"** | | **2 to 13 msec.** |
| **- from "1" to "0"** | | **10 to 35 msec.** |
| **Cable length** | | |
| **- shielded** | | **1000 m (3281 ft.)** |
| **- unshielded** | | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage [1)]** | | |
| **(between groups)** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V AC** |
| **Rated isolation voltage** | | |
| **(L1 to ⏚ )** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V AC** |
| **Connection of 2-wire** | | |
| **BERO proximity switches** | | **possible** |
| **- quiescent current** | | **3 mA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **5 mA** |
| **Simultaneity factor** | | |
| **(per group, for L1=264 V)** | | |
| **- at 25 °C** | | **100 %** |
| **- at 55 °C** | | **75 %** |
| **Power losses of** | | |
| **the module** | **typically** | **11 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

**[1] Connection of different phases is not permissible.**

## Digital Input Module 16 x 230 V UC (6ES5 436-7LB11)




**Terminal Assignment**



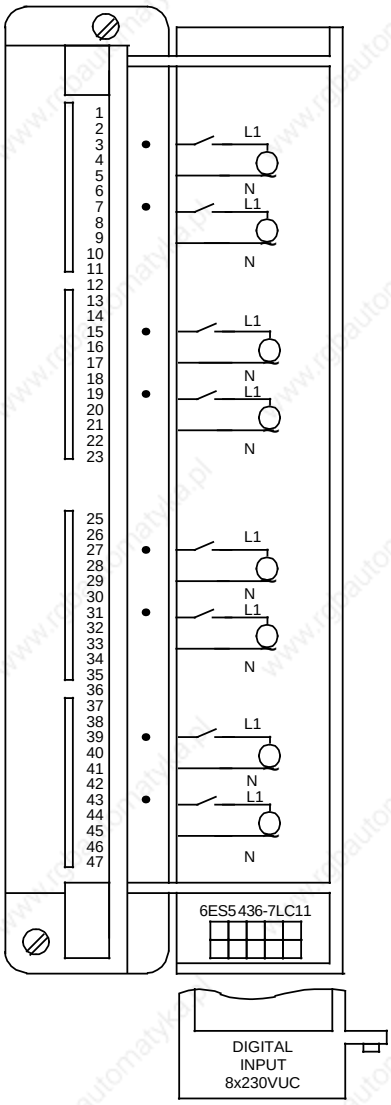

**Block diagram**

**Technical Specifications**

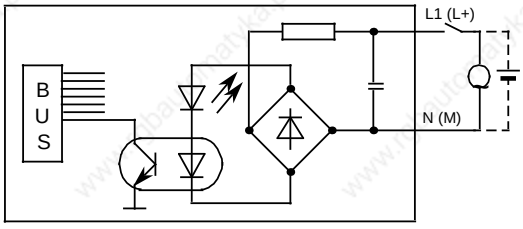
| | | |
|---|---|---|
| **Number of inputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **2** |
| **Input voltage L1** | | |
| **- rated value** | | **230 V UC** |
| **- frequency** | | **47 to 63 Hz** |
| **- for "0" signal** | | **0 to 70 V** |
| **- for "1" signal** | | **170 to 264 V** |
| **Input current** | | |
| **at "1" signal** | | |
| **- for AC, 50 Hz** | **typically** | **15mA** |
| **- for DC** | **typically** | **2.2 mA** |
| **Response time** | | |
| **- from "0" to "1"** | | **2 to 13 msec.** |
| **- from "1" to "0"** | | **10 to 35 msec.** |
| **Cable length** | | |
| **- shielded** | | **1000 m (3281 ft.)** |
| **- unshielded** | | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage [1]** | | |
| **(between groups)** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V** |
| **Rated isolation voltage** | | |
| **(L1 to ⏚ )** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V** |
| **Connection of 2-wire** | | |
| **BERO proximity switches** | | **possible** |
| **- quiescent current** | | **3 mA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **5mA** |
| **Simultaneity factor** | | |
| **(per group, for L1=264 V)** | | |
| **- at 25 °C** | | **100 %** |
| **- at 55 °C** | | **75 %** |
| **Power losses of** | | |
| **the module** | **typically** | **11 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

**[1] Connection of different phases is not permissible.**

## Digital Input Module 8 x 230 V UC (6ES5 436-7LC11)




**Terminal Assignment**




**Block diagram**

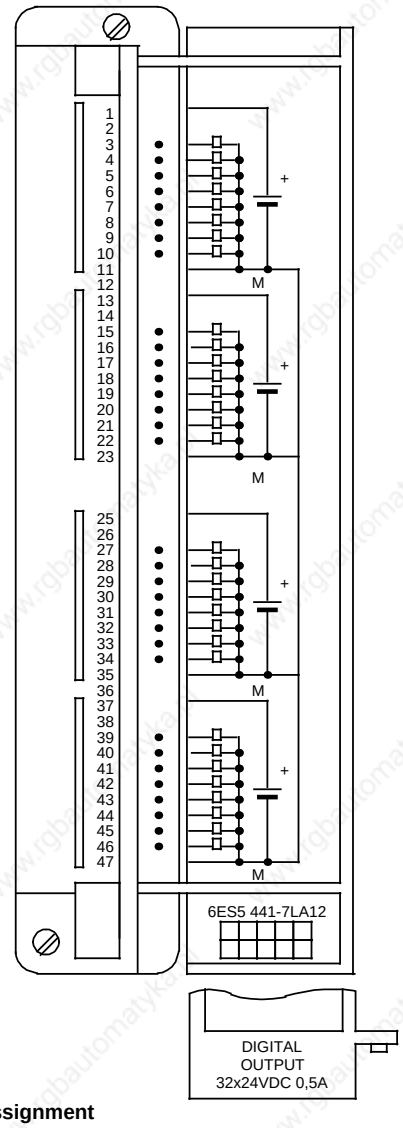
**Technical Specifications**

| | | |
|---|---|---|
| **Number of inputs** | | **8** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **1** |
| **Input voltage L1** | | |
| **- rated value** | | **230 V UC** |
| **- frequency** | | **47 to 63 Hz** |
| **- for "0" signal** | | **0 to 100 V** |
| **- for "1" signal** | | **170 to 264 V** |
| **Input current at** | | |
| **"1" signal AC** | **typically** | **16 mA** |
| **DC** | **typically** | **2.2 mA** |
| **Response time** | | |
| **- from "0" to "1"** | | **2 to 13 msec.** |
| **- from "1" to "0"** | | **10 to 25 msec.** |
| **Cable length** | | |
| **- shielded** | | **1000 m (3281 ft.)** |
| **- unshielded** | | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage [1)]** | | |
| **(between groups)** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **2700 V** |
| **Rated isolation voltage** | | |
| **(L1 to ⏚ )** | | **250 V** |
| **- isolation group** | | **C** |
| **- tested with** | | **2700 V** |
| **Connection of 2-wire** | | |
| **BERO proximity switches** | | **possible** |
| **- quiescent current** | | **5 mA** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **5 mA** |
| **Power losses of** | | |
| **the module** | **typically** | **5 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

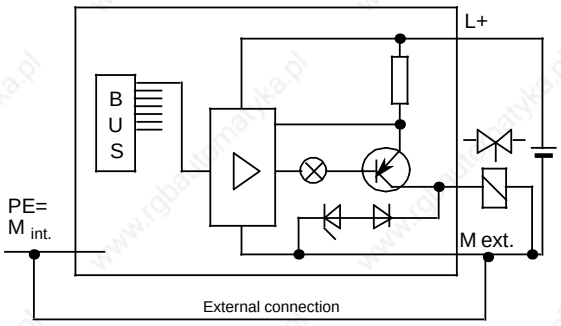
**1** **Connection of different phases is not permissible.**

## 15.2.5 Digital Output Modules

### Digital Output Module 32 x 24 V DC; 0.5 A, Nonfloating (6ES5 441-7LA12)



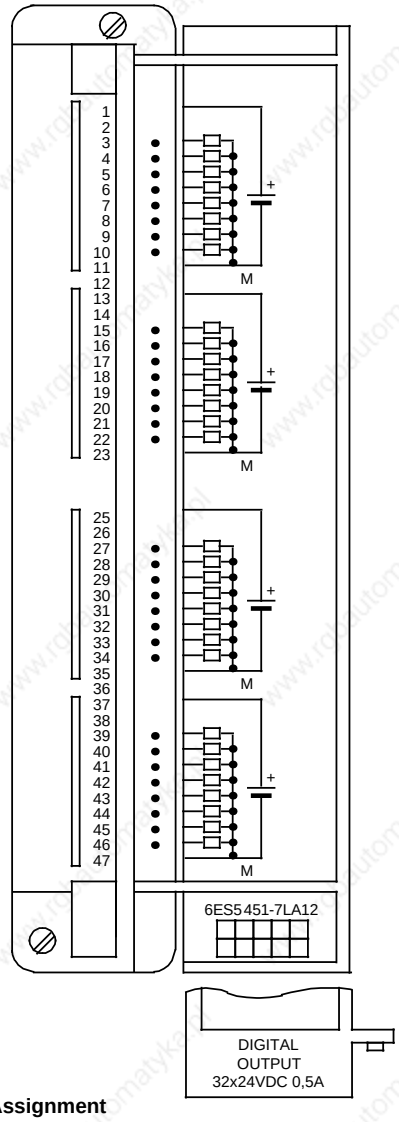

**Terminal Assignment**



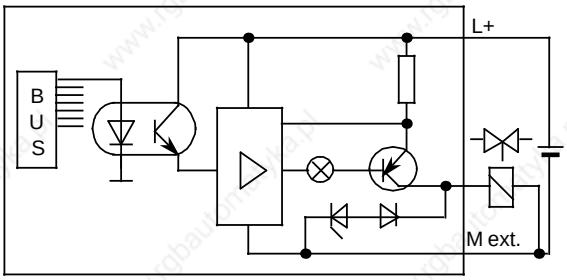

**Block diagram**

| Technical Specifications | | |
|---|---|---|
| Number of outputs | | 32 |
| Floating | | no |
| Load voltage L+ | | |
| - rated value | | 24 V DC |
| - permissible range | | 20 to 30 V |
| - surge voltage at t 0.5 sec. | | 35 V |
| Output voltage | | |
| - at "1" signal | minimum | L+- 2.5 V |
| Output current | | |
| at "1" signal | | |
| - rated value | | 0.5 A |
| - lamp load | maximum | 5 W |
| Leakage current | | |
| at "0" signal | maximum | 1 mA |
| Parallel connection | | |
| of outputs | | not possible |
| Permissible total current | | |
| of outputs | | 100 % at 25 °C and 50 % at 55 °C (related to the sum of currents) |
| Short circuit protection | | electronic |
| Limitation of the voltage | | |
| induced on circuit interruption | | - 15 V |
| Switching frequency | | |
| - inductive load | maximum | 0.5 Hz |
| - resistive load | maximum | 100 Hz |
| Cable length | | |
| - shielded | maximum | 1000 m (3281 ft.) |
| - unshielded | maximum | 600 m (1969 ft.) |
| Isolation rating | | according to VDE 0160 |
| Current consumption | | |
| - from 5 V (internal) | | 10 mA |
| - from L+(without load) | | 17 mA/per group |
| Power losses of | | |
| the module | typically | 20 W |
| Weight | approx. | 0.7 kg (1.54 lb.) |

## Digital Output Modules 32 x 24 V DC; 0.5 A, Floating (6ES5 451-7LA12)




**Terminal Assignment**

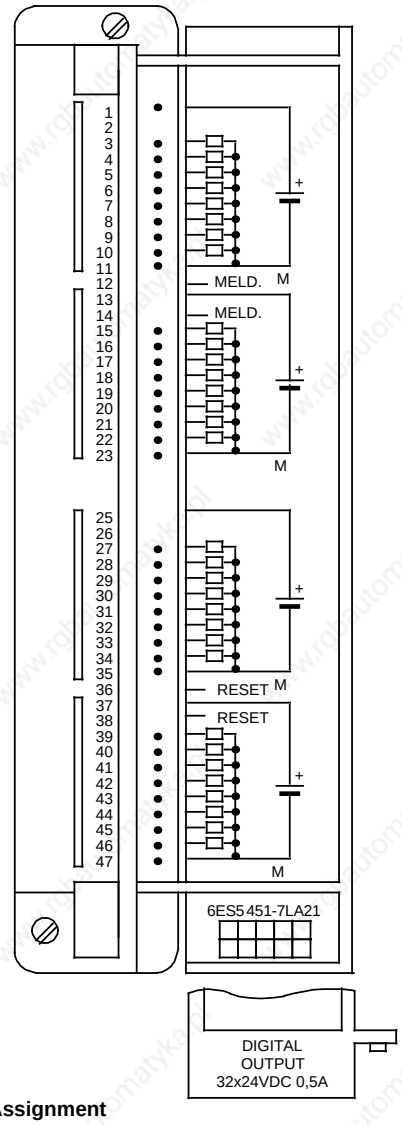



**Simplified Schematic**

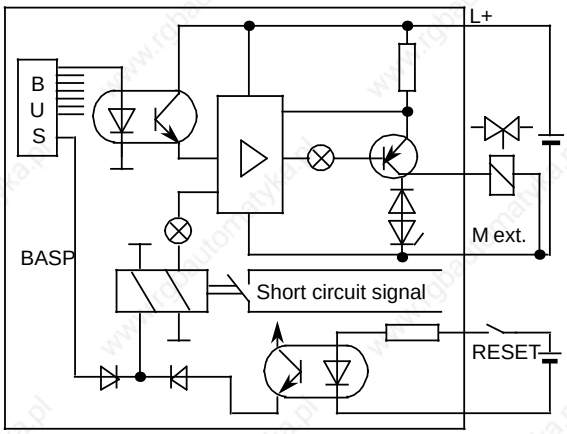
**Technical Specifications**

| | | |
|---|---|---|
| **Number of outputs** | | **32** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **8** |
| **Load voltage L+** | | |
| **- rated value** | | **24 V DC** |
| **- permissible range** | | **20 to 30 V** |
| **- surge voltage at t 0.5 sec.** | | **35 V** |
| **Output voltage** | | |
| **- at "1" signal** | **minimum** | **L+- 2.5 V** |
| **Output current for "1" signal** | | |
| **- rated value** | | **0.5 A** |
| **- lamp load** | **maximum** | **5 W** |
| **Leakage current at "0" signal** | **maximum** | **1 mA** |
| **Parallel connection of outputs** | | **not possible** |
| **Permissible total current of outputs** | | **100 % at 25 °C and 50 % at 55 °C (related to the sum of currents)** |
| **Short circuit protection** | | **electronic** |
| **Limitation of the voltage induced on circuit interruption** | | **- 15 V** |
| **Switching frequency** | | |
| **- inductive load** | **maximum** | **0.5 Hz** |
| **- resistive load** | **maximum** | **100 Hz** |
| **Cable length** | | |
| **- shielded** | | **1000 m (3281 ft.)** |
| **- unshielded** | | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage (between groups)** | | **30 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Rated isolation voltage (L+ to ⏚ )** | | **30 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **100 mA** |
| **- from L+(without load)** | | **17 mA/per group** |
| **Power losses of the module** | **typically** | **20 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

## Digital Output Module 32 x 24 V DC; 0.5 A, Floating (6ES5 451-7LA21)




**Terminal Assignment**



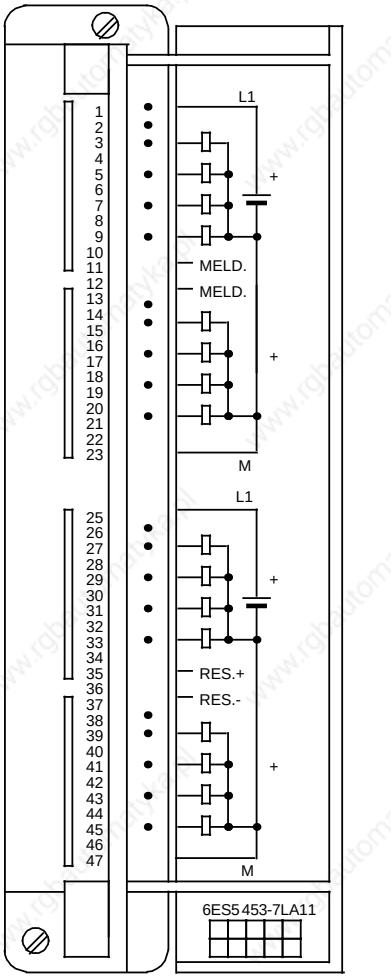

**Block diagram**

**Technical Specifications**

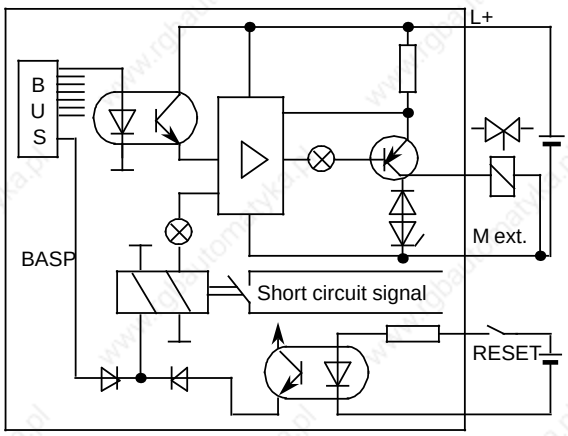
| | | |
|---|---|---|
| **Number of outputs** | | **32** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **8** |
| **Load voltage L+** | | |
| **- rated value** | | **24 V DC** |
| **- permissible range** | | **20 to 30 V** |
| **- surge voltage at t 0.5 sec.** | | **35 V** |
| **Output voltage** | | |
| **- for "1" signal** | **minimum** | **L+- 2.5 V** |
| **Output current for "1" signal** | | |
| **- rated value** | | **0.5 A** |
| **- lamp load** | **maximum** | **5 W** |
| **Leakage current at "0" signal** | **maximum** | **1 mA** |
| **Parallel connection of outputs** | | **not possible** |
| **Permissible total current of outputs** | | **100 % at 25 °C and 50 % at 55 °C (related to the sum of currents)** |
| **Short circuit protection** | | **electronic** |
| **Short circuit indicator** | | **red LED (per group)** |
| **Short circuit signal (latching relay contact)** | | **latching[1]** |
| **Type of relay** | | **TQ2SA-L2-5V-Z (Nais Company)** |
| **- load** | | **30V DC; 0.1 A** |
| **- switching capacity** | | **3 W or 6 VA** |
| **- reset input** | | **24V DC** |
| **Limitation of the voltage induced on circuit interruption** | | **- 15 V** |
| **Switching frequency** | | |
| **- inductive load** | **maximum** | **0.5 Hz** |
| **- resistive load** | **maximum** | **100 Hz** |
| **Cable length** | | |
| **- shielded** | | **1000 m (3281 ft.)** |
| **- unshielded** | | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage (between groups)** | | **30 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Rated isolation voltage (L+ to ⏚ )** | | **30 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **100 mA** |
| **- from L+(without load)** | | **17 mA/per group** |
| **Power losses of the module** | **typically** | **20 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

**[1] Pickup delay: approx. 1 sec. after start of short circuit**

## Digital Output Module 16 x 24 to 60 V DC; 0.5 A, Floating (6ES5 453-7LA11)




**Terminal Assignment**



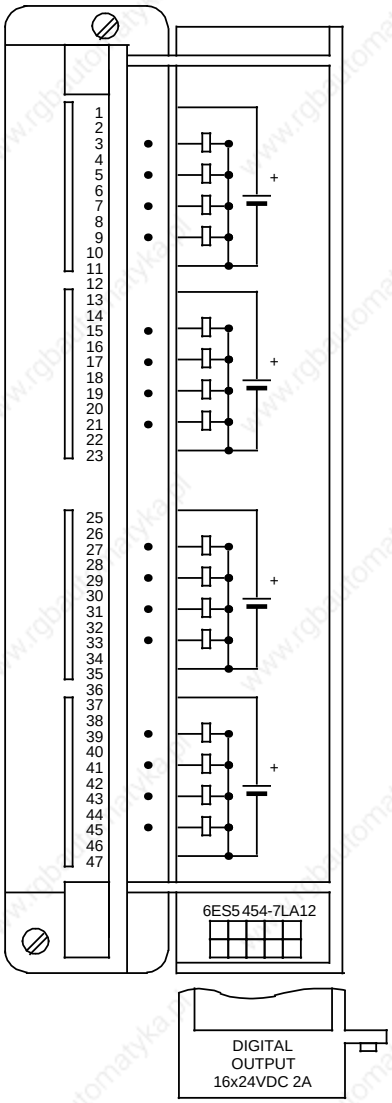

**Block diagram**

**Technical Specifications**

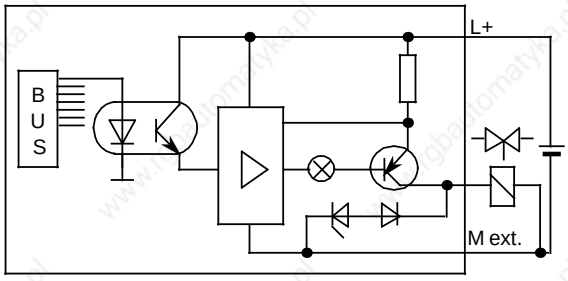
| | | |
|---|---|---|
| **Number of outputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **8** |
| **Load voltage L+** | | |
| **- rated value** | | **24 to 60 V DC** |
| **- permissible range** | | **20 to 75 V** |
| **- surge voltage at t 0.5 sec.** | | **87 V** |
| **Ouput voltage** | | |
| **- for "1" signal** | **maximum** | **L+- 2.5 V** |
| **Output current for "1" signal** | | |
| **- rated value** | | **0.5 A** |
| **- lamp load** | **maximum** | **5 W** |
| **Leakage current at "0" signal** | **maximum** | **1 mA** |
| **Parallel connection of outputs** | | **not possible** |
| **Permissible total current of outputs, per group** | | **100 %** |
| **Short circuit protection** | | **electronic** |
| **Short circuit indicator** | | **red LED (each group)** |
| **Short circuit signal (latching relay contact)** | | **latching [1]** |
| **Type of relay** | | **V23042 B201 B101** |
| **- load** | | **60 V DC; 0.2 A** |
| **- switching capacity** | | **20 W or 35 VA** |
| **- reset input** | | **24 V to 60 V DC** |
| **Limitation of the voltage induced on circuit interruption** | | **- 30 V** |
| **Switching frequency** | | |
| **- inductive load** | **maximum** | **0.5 Hz** |
| **- resistive load** | **maximum** | **100 Hz** |
| **Cable length** | | |
| **- shielded** | **maximum** | **1000 m (3281 ft.)** |
| **- unshielded** | **maximum** | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage (between groups)** | | **75 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Rated isolation voltage (L+ to ⏚ )** | | **75 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Current consumption** | | |
| **- from +5 V (internal)** | | **50 mA** |
| **- from L+(without load)** | | **50 mA/per group** |
| **Power losses of the module** | **typically** | **14 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

**[1] Pickup delay: approx. 1 sec. after start of short circuit**

## Digital Output Module 16 x 24 V DC; 2 A, Floating (6ES5 454-7LA12)




**Terminal Assignment**

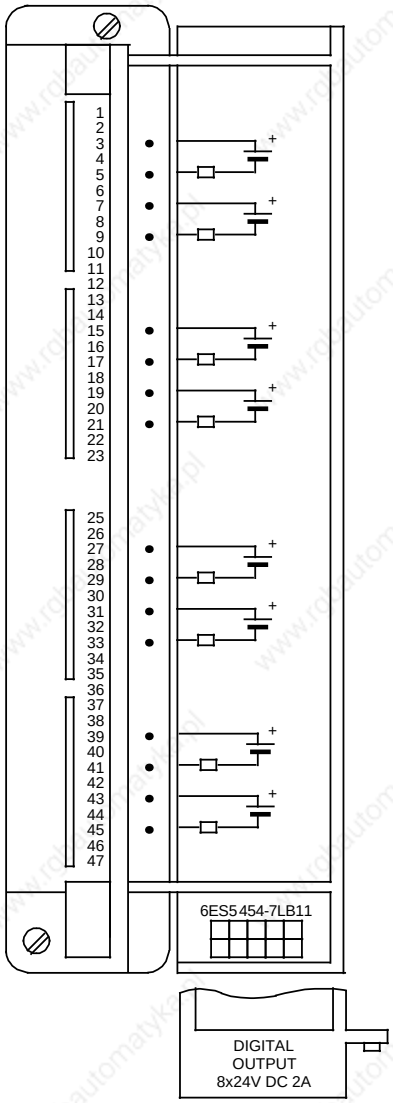



**Block diagram**

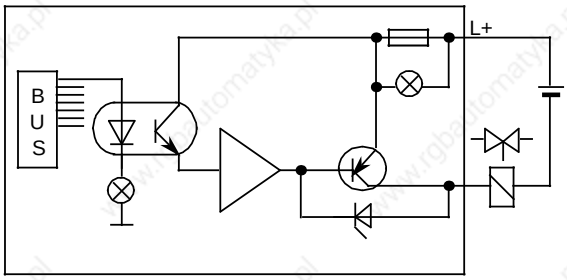
**Technical Specifications**

| | | |
|---|---|---|
| **Number of outputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **4** |
| **Load voltage L+** | | |
| **- rated value** | | **24 V DC** |
| **- permissible range** | | **20 to 30 V** |
| **- surge voltage at t 0.5 sec.** | | **35 V** |
| **Output voltage** | | |
| **- for "1" signal** | **minimum** | **L+- 3 V** |
| **Output current** | | |
| **at"1" signal** | | |
| **- rated value** | | **2 A** |
| **- lamp load** | **maximum** | **10 W** |
| **Leakage current** | | |
| **at "0" signal** | **maximum** | **1 mA** |
| **Parallel connection** | | |
| **of outputs** | | **not possible** |
| **Permissible total current** | | |
| **of outputs, per group** | | **50 % (related to the sum of the currents of a group)** |
| **Short circuit protection** | | **electronic** |
| **Limitation of the voltage induced** | | |
| **on circuit interruption** | | **- 15 V** |
| **Switching frequency** | | |
| **- inductive load** | **maximum** | **0.27 Hz** |
| **- resistive load** | **maximum** | **100 Hz** |
| **Cable length** | | |
| **- shielded** | **maximum** | **1000 m (3281 ft.)** |
| **- unshielded** | **maximum** | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage** | | |
| **(between groups)** | | **30 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Rated isolation voltage** | | |
| **(L+ to ⏚ )** | | **30 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **50 mA** |
| **- from L+(without load)** | | **8.5 mA/per group** |
| **Power losses of** | | |
| **the module** | **typically** | **20 W** |
| **Weight** | **approx.** | **1.1 kg (2.43 lb.)** |

## Digital Output Module 8 x 24 V DC; 2 A, Floating (6ES5 454-7LB11)




**Terminal Assignment**

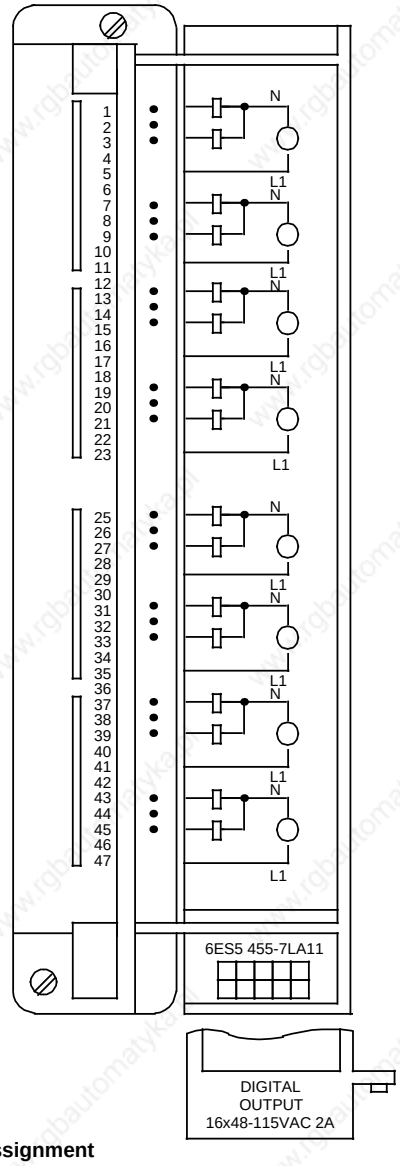



**Block diagram**

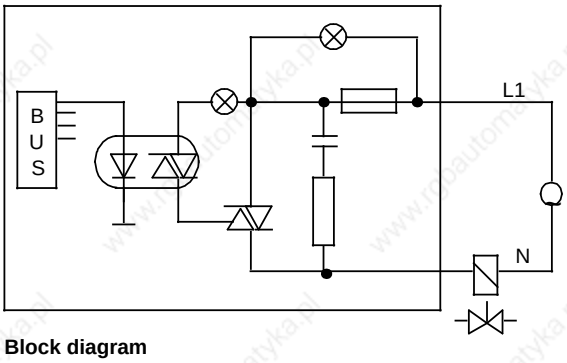
**Technical Specifications**

| | | |
|---|---|---|
| **Number of outputs** | | **8** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **1** |
| **Load voltage L+** | | |
| **- rated value** | | **24 V DC** |
| **- permissible range** | | **20 to 30 V** |
| **- surge voltage at t 0.5 sec.** | | **35 V** |
| **Output voltage** | | |
| **- for "1" signal** | **minimum** | **L+- 3 V** |
| **Output current** | | |
| **- rated value** | | **2 A** |
| **- lamp load** | **maximum** | **10 W** |
| **Leakage current** | | |
| **at "0" signal** | **maximum** | **1 mA** |
| **Parallel connection of outputs** | | **possible** |
| **- maximum current** | | **1 x rated current** |
| **Permissible total current of outputs, per group** | | **100 % at 25 °C and 50 % at 55 °C (related to the sum of the currents of a group)** |
| **Short circuit protection (for each group)** | | **with F 2.5 A fuse (e.g. Wickmann 19340)** |
| **Limitation of the voltage induced on circuit interruption** | **typically** | **- 23 V** |
| **Switching frequency** | | |
| **- inductive load** | **maximum** | **0.27 Hz** |
| **- resistive load** | **maximum** | **100 Hz** |
| **Cable length** | | |
| **- shielded** | **maximum** | **1000 m (3281 ft.)** |
| **- unshielded** | **maximum** | **600 m (1969 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage (between groups)** | | **30 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Rated isolation voltage (L+ to ⏚ )** | | **30 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Current consumption** | | |
| **- from + 5 V (internal)** | **maximum** | **50 mA** |
| **Power losses of the module** | **typically** | **20 W** |
| **Weight** | **approx.** | **0.8 kg (1.76 lb.)** |

## Digital Output Module 16 x 48 to 115 V AC;2 A, Floating (6ES5 455-7LA11)




**Terminal Assignment**

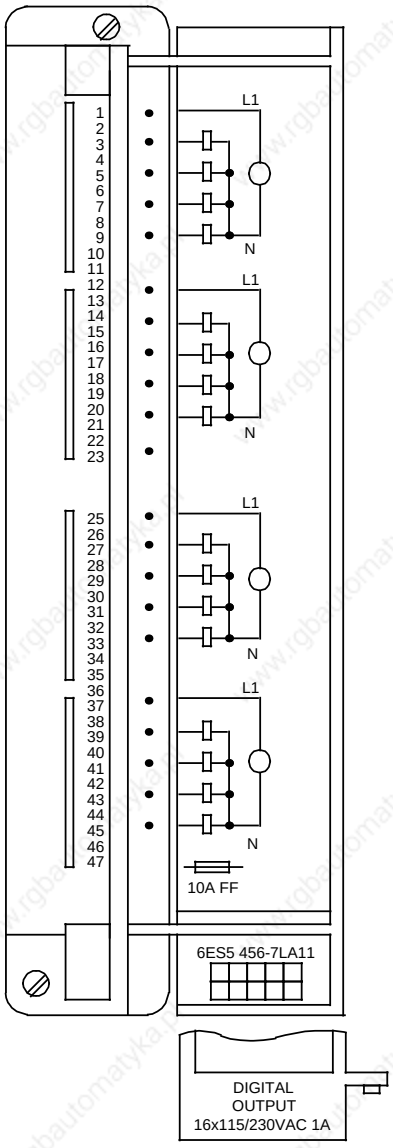



**Block diagram**

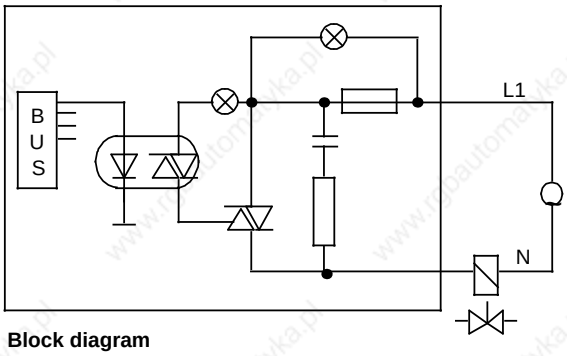
**Technical Specifications**

| | | |
|---|---|---|
| **Number of outputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **2** |
| **Load voltage L1** | | |
| **- rated value** | | **48/115 V AC** |
| **- frequency** | | **47 to 63 Hz** |
| **- permissible range** | | **40 to 140 V** |
| **Output voltage** | | |
| **- for "1" signal** | **minimum** | **L1 - 7 V** |
| **Output current** | | |
| **at "1" signal** | | |
| **- rated value** | | **2 A/per group** |
| **- permissible range** | | **40 mA to 2 A** |
| **- lamp load** | **maximum** | **50/100 W/per group** |
| **Leakage current** | | |
| **at "0" signal** | **maximum** | **1/3 mA** |
| **Parallel connection of** | | |
| **outputs** | | **not possible** |
| **Making capacity** | | **depends on the size of the fuse** |
| **Permissible total current** | | |
| **of outputs** | | **100 %** |
| **Short circuit protection** | | **with Gould GAB4 fuse** |
| **(for each group)** | | **or Bussmann ABC4** |
| **Fault indication** | | **defective fuse** |
| **(red LED for each group)** | | |
| **Switching frequency** | **maximum** | **10 Hz** |
| **Cable length** | | |
| **- shielded** | **maximum** | **1000 m (3281 ft.)** |
| **- unshielded** | **maximum** | **300 m (984 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage** | | |
| **(between groups)** | | **250 V AC** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V AC** |
| **Rated isolation voltage** | | |
| **(L1 to ⏚ )** | | **250 V AC** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V AC** |
| **Current consumption** | | |
| **- from 5 V (internal)** | **maximum** | **175 mA** |
| **Power losses of** | | |
| **the module** | **typically** | **16 W** |
| **Weight** | **approx.** | **1.1 kg (2.43 lb.)** |

## Digital Output Module 16 x 115 to 230 V AC; 1 A, Floating (6ES5 456-7LA11)




**Terminal Assignment**



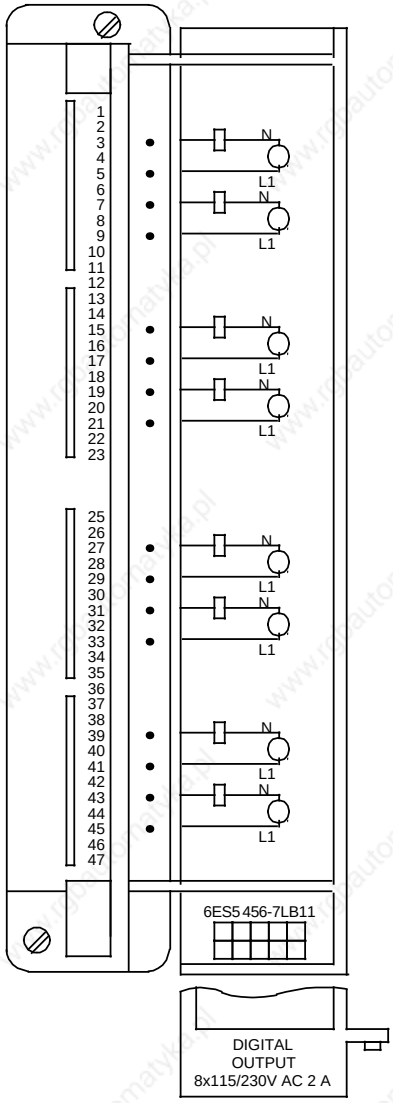

**Block diagram**

**Technical Specifications**

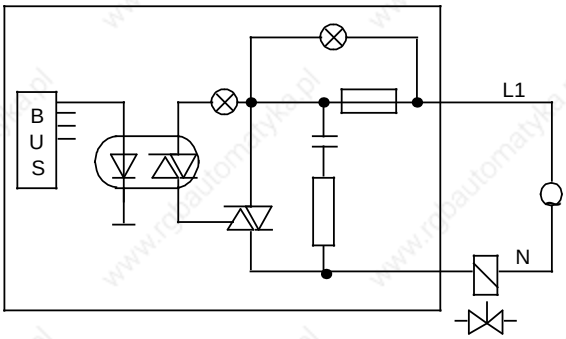
| | | |
|---|---|---|
| **Number of outputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **4** |
| **Load voltage L1** | | |
| **- rated value** | | **115/230 V AC** |
| **- frequency** | | **47 to 63 Hz** |
| **- permissible range** | | **89 to 264 V** |
| **Output voltage** | | |
| **- at "1" signal** | **minimum** | **L1 - 7 V** |
| **Output current** | | |
| **at "1" signal** | | |
| **- rated value** | | **1 A** |
| **- permissible range** | | **40 mA to 1 A** |
| **- lamp load** | | **25/50 W** |
| **Leakage current** | | |
| **at "0" signal** | **typically** | **3/5 mA [1]** |
| **Parallel connection of outputs** | | **not possible** |
| **Making capacity** | | **depends on the size of the fuse** |
| **Permissible total current of outputs** | | **100 %** |
| **Short circuit protection (for each group)** | | **fuse (10 A FF) (e.g. Wickmann 19231)** |
| **Fault indication (red LED for each group)** | | **defective fuse** |
| **Switching frequency** | **maximum** | **10 Hz** |
| **Cable length** | | |
| **- shielded** | | **1000 m (3281 ft.)** |
| **- unshielded** | | **300 m (984 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage (between groups)** | | **250 V AC** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V AC** |
| **Rated isolation voltage (L1 to ⏚ )** | | **250 V AC** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V AC** |
| **Current consumption** | | |
| **- from 5 V (internal)** | **maximum** | **70 mA** |
| **Power losses of the module** | **typically** | **16 W** |
| **Weight** | **approx.** | **1.1 kg (2.43 lb.)** |

**[1] Please note the max. dropout capacity of the load connected (contactors of the 3TI1 to 3TI5 series and contactors of the SIMICOMT series cannot be triggered)!**

## Digital Output Module 8 x 115 to 230 V AC; 2 A (6ES5 456-7LB11)




**Terminal Assignment**

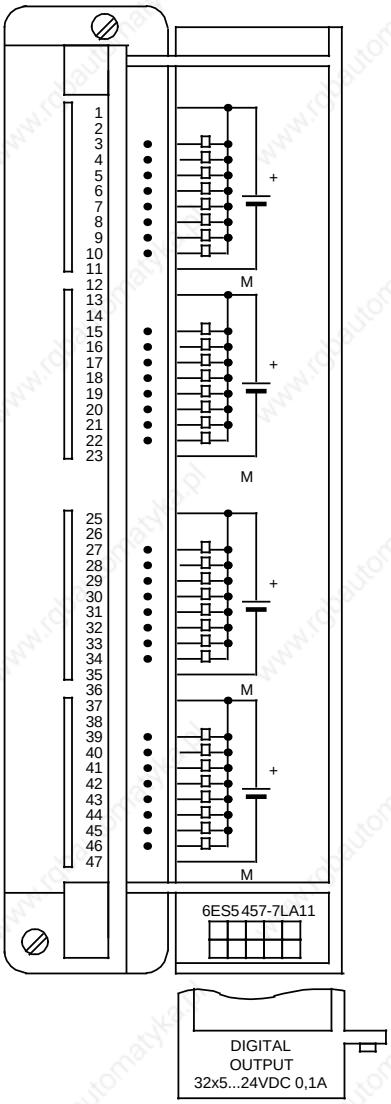



**Block diagram**

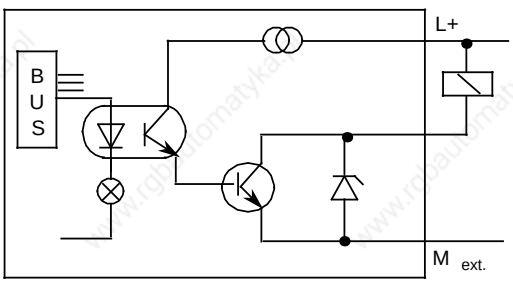
| Technical Specifications | | |
|---|---|---|
| Number of outputs | | 8 |
| Floating | | yes (optocoupler) |
| - in groups of | | 1 |
| Load voltage L1 | | |
| - rated value | | 115 to 230 V AC |
| - frequency | | 47 to 63 Hz |
| - permissible range | | 89 to 264 V |
| Output voltage | | |
| - at "1" signal | minimum | L1 - 7 V |
| Output current at "1" signal | | |
| - rated value | | 2 A |
| - permissible range | | 40 mA to 2 A |
| - lamp load | | 25/50 W |
| Output current at "0" signal | typically | 3 to 5 mA [1] |
| Parallel connection of outputs | | not possible |
| Making capacity | | depends on the size of the fuse |
| Permissible total current of outputs | | 100 % |
| Short circuit protection (for each group) | | fuse (6.3 A FF) (e.g. Wickmann 19231) |
| Fault indication (red LED for each group) | | defective fuse |
| Switching frequency | maximum | 10 Hz |
| Cable length | | |
| - shielded | | 1000 m |
| - unshielded | | 300 m |
| Isolation rating | | according to VDE 0160 |
| Rated isolation voltage (between groups) | | 250 V AC |
| - isolation group | | C |
| - tested with | | 2700 V AC |
| Rated isolation voltage (L1 to ⏚) | | 250 V AC |
| - isolation group | | C |
| - tested with | | 2700 V AC |
| Current consumption | | |
| - from 5 V (internal) | maximum | 35 mA |
| Power losses of the module | typically | 16 W |
| Weight | approx. | 1.1 kg (2.43 lb.) |

[1] Please note the max. dropout capacity of the load connected (contactors of the 3TI1 to 3TI5 series and contactors of the SIMICOMT series cannot be triggered)!

## Digital Output Module 32 x 5 to 24 V DC; 0.1 A, Floating (6ES5 457-7LA11)




**Terminal Assignment**



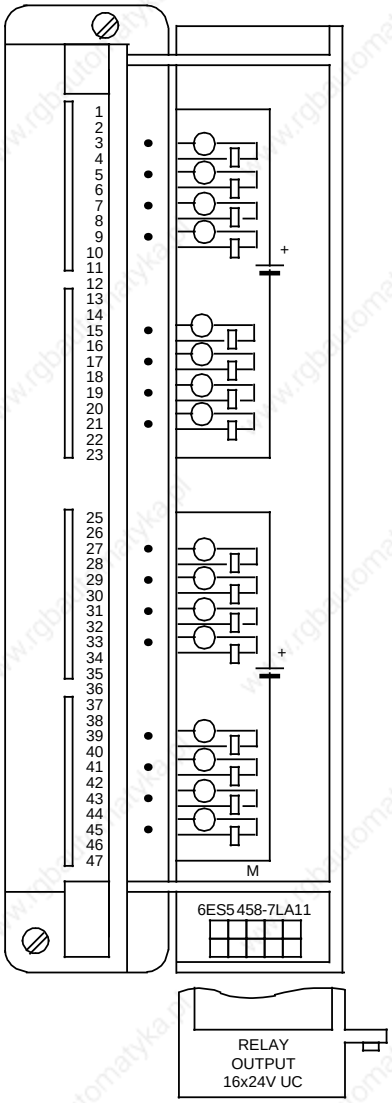

**Block diagram**

**Technical Specifications**

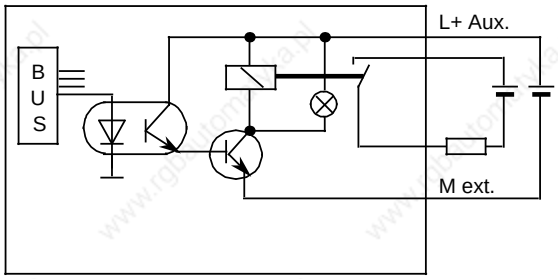
| | | |
|---|---|---|
| Number of outputs | | 32 |
| Floating | | yes (optocoupler) |
| - in groups of | | 8 |
| Load voltage L1 | | |
| - rated value | | 5 to 24 V DC |
| - permissible range | | 4.75 to 30 V |
| Output voltage [1] | | TTL compatible |
| Output current for "1" signal | maximum | 100 mA |
| Parallel connection of outputs | | possible |
| Permissible total current of outputs | | 100 % |
| Short circuit protection | | not available |
| Limitation of the voltage induced on circuit interruption (for $V_p$=30 V) | | - 10 V |
| Switching frequency | | |
| - inductive load | | 2 Hz |
| - resistive load | | 10 Hz |
| Cable length | | |
| - shielded | | 1000 m (3281 ft.) |
| - unshielded | | 300 m (984 ft.) |
| Isolation rating | | according to VDE 0160 |
| Rated isolation voltage (between groups) | | 30 V |
| - isolation group | | C |
| - tested with | | 500 V |
| Rated isolation voltage (5 V to ⏚ ) | | 30 V |
| - isolation group | | C |
| - tested with | | 500 V |
| Current consumption | | |
| - from 5 V (internal) | maximum | 100 mA |
| - from L1 (without load) | maximum | 4 mA |
| Power losses of the module | typically | 6 W |
| Weight | approx. | 0.7 kg (1.54 lb.) |

[1] Transistor with open collector - current sinking

## Relay Output Module for Measuring Currents 16 x 24 V DC (6ES5 458-7LA11)




**Terminal Assignment**

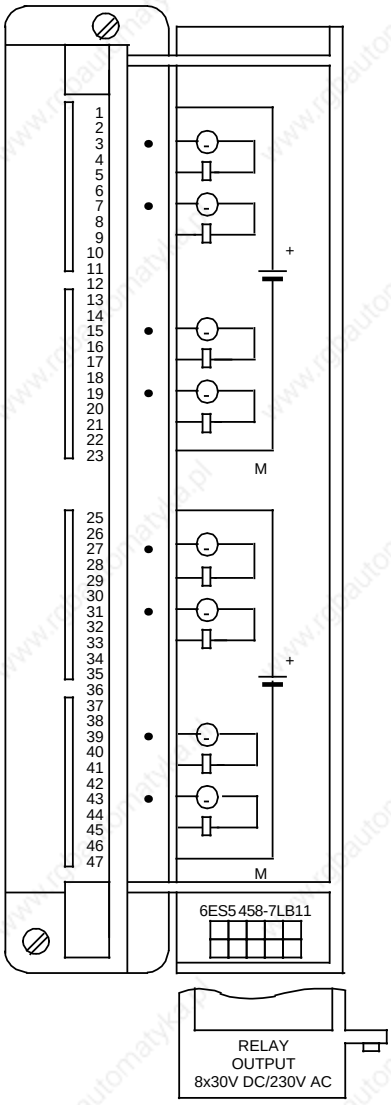



**Block diagram**

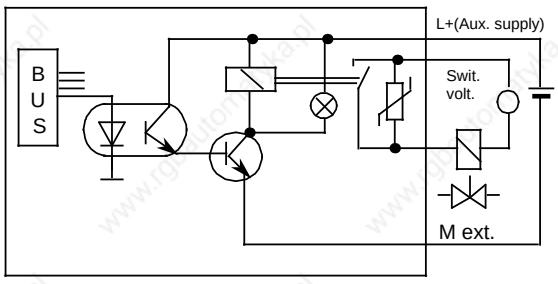
**Technical Specifications**

| | | |
|---|---|---|
| **Number of outputs** | | **16** |
| **- contact bridge** | | **no** |
| **- galvanic isolation** | | **yes** |
| **- in groups of** | | **1** |
| **- relay type** | | **3700-2501-011 (Günther)** |
| **Limiting continuous current per contact** | | **0.5 A** |
| **Parallel connection of outputs** | | **possible** |
| **Permissible total current of outputs** | | **100 %** |
| **Switching frequency** | | |
| **- resistive load** | **maximum** | **60 Hz** |
| **- inductive load** | | **not permissible** |
| **Switching voltage** | **maximum** | **30 V DC** |
| **Switching capacity of the contacts** | | |
| **- resistive load** | | **10 W at 0.5 A;** |
| **- inductive load** | | **not permissible** |
| **Number of contact operating cycles according to VDE 0660, Part 200** | | |
| **- DC 11** | | **1 x $10^9$** |
| **Supply voltage L+ (for relay)** | | |
| **- rated value** | | **24 V DC** |
| **- permissible range** | | **20 to 30 V** |
| **- surge voltage at t<0.5 sec.** | | **35 V** |
| **- ripple** | **maximum** | **3.6 V** |
| **Cable length** | | |
| **- shielded** | | **1000 m (3281 ft.)** |
| **- unshielded** | | **300 m (984 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage (between contacts)** | | **30 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Rated isolation voltage (contacts to L+)** | | **30 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Rated isolation voltage (contacts to ⏚ )** | | **30 V DC** |
| **- isolation group** | | **C** |
| **- tested with** | | **500 V AC** |
| **Current consumption** | | |
| **- from 5 V (internal)** | **maximum** | **50 mA** |
| **- fromL+(for relay)** | | **240 mA** |
| **Power losses of the module** | **typically** | **5W** |
| **Weight** | **approx.** | **0.8 kg (1.76 lb.)** |

## Relay Output Module 8 x 30 V DC/230 V AC (6ES5 458-7LB11)




**Terminal Assignment**

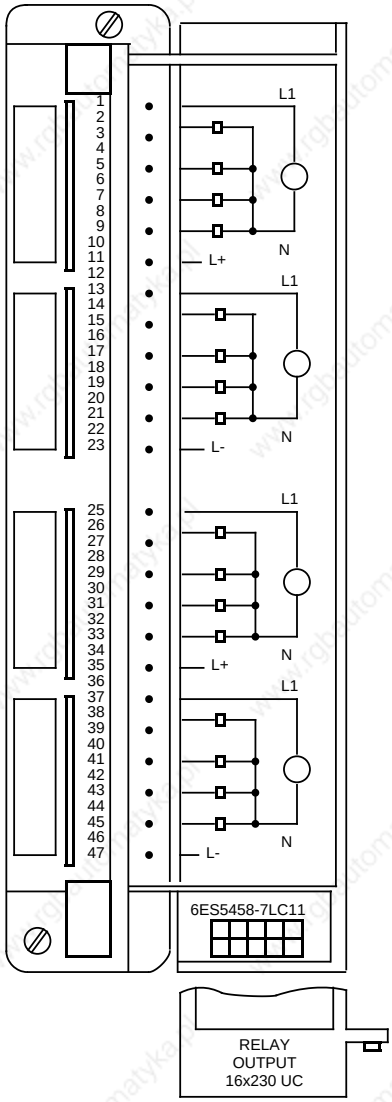



**Block diagram**

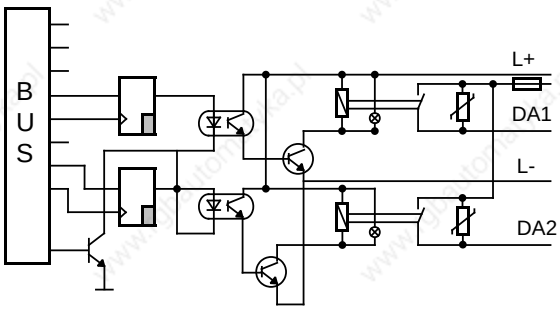
**Technical Specifications**

| | | |
|---|---|---|
| **Number of outputs** | | **8** |
| **- contact bridge** | | **varistor SIOV-S07-K275** |
| **- floating** | | **yes** |
| **- in groups of** | | **1** |
| **- relay type** | | **V23057-A006-A402 (Siemens)** |
| **Limiting continuous current per contact** | | **5 A** |
| **Parallel connection of outputs** | | **possible** |
| **Permissible total current of outputs** | | **100 %** |
| **Switching capacity of the contacts** | | |
| **- resistive load** | | **5 A at 250 V AC<br>2.5 A at 30 V DC** |
| **- inductive load** | | **1.5 A at 250 V AC<br>0.5 A at 30 V DC** |
| **Switching frequency** | **maximum** | **10 Hz** |
| **Number of contact operating cycles according to VDE 0660, Part 200** | | |
| **- AC 11** | | **1.5 x $10^6$** |
| **- DC 11** | | **0.5 x $10^6$** |
| **Supply voltage L+ (for relay)** | | |
| **- rated value** | | **24 V DC** |
| **- permissible range** | | **20 to 30 V** |
| **- surge voltage at t** | **0.5 sec.** | **35 V** |
| **- ripple** | **maximum** | **3.6 V** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage (between contacts)** | | **250 V AC** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V AC** |
| **Rated isolation voltage (between contacts L+)** | | **250 V AC** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V AC** |
| **Rated isolation voltage (between contacts ⏚ )** | | **250 V AC** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V AC** |
| **Current consumption** | | |
| **- from 5 V (internal)** | **maximum** | **50 mA** |
| **- from L+(for relay)** | | **200 mA** |
| **Power losses of the module** | **typically** | **4 W** |
| **Weight** | **approx.** | **0.8 kg (1.76 lb.)** |

## Relay Output Module 16 x 230 V UC (6ES5 458-7LC11)




**Terminal assignment**

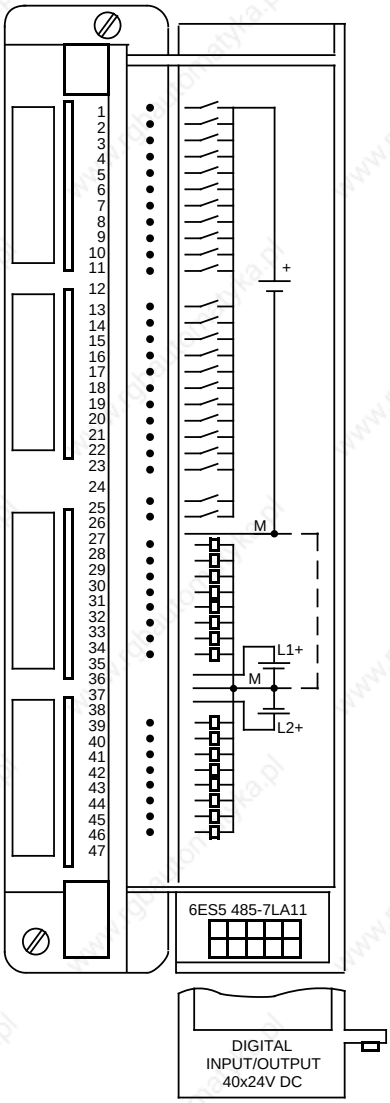



**Block diagram**

**Technical Specifications**

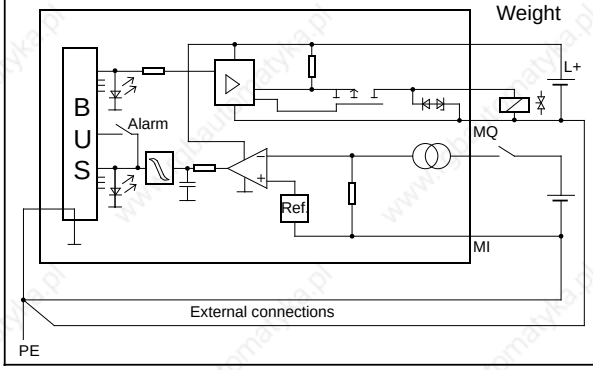
| | | |
|---|---|---|
| **Number of outputs** | | **16** |
| **- contact bridge** | | **SIOV-S07-K275 varistor** |
| **- galvanic isolation** | | **yes** |
| **- in groups of** | | **4 outputs** |
| **- relay type** | | **MSR V23061-B1007-A401** |
| **Switching capacity of the contacts** | | |
| **- resistive load** | | **5.0 A at 250 V AC<br>5.0 A at 30 V DC<br>0.4 A at 110 V DC** |
| **- inductive load** | | **1.5 A at 250 V AC<br>1.0 A at 30 V DC<br>0.08 A at 110 V DC** |
| **Switching frequency** | | |
| **- resistive load** | **maximum** | **10 Hz** |
| **- inductive load** | | **2 Hz** |
| **Current load of the relay contacts** | | |
| **- one relay per root** | | **5 A/contact** |
| **- two relays " "** | | **4 A/contact** |
| **- three " " "** | | **2.5 A/contact** |
| **- four " " "** | | **2 A/contact** |
| **Number of contact operating cycles according to VDE 0660, Part 200** | | |
| **- DC 11** | | **1.5 x $10^6$ (AC)<br>0.5 x $10^6$ (DC)** |
| **Supply voltage L+/L- (for relay)** | | |
| **- rated value** | | **24 V DC** |
| **- permissible range** | | **20 to 30 V** |
| **- surge voltage at t 0.5 sec.** | | **35 V** |
| **- ripple** | **maximum** | **3.6 V** |
| **Cable length** | | |
| **- shielded** | | **1000 m (3281 ft.)** |
| **- unshielded** | | **300 m (984 ft.)** |
| **Isolation rating** | | **according to VDE 0160** |
| **Rated isolation voltage (contacts to L+)** | | **250 V AC** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V AC** |
| **Rated isolation voltage (contacts to ⏚ )** | | **250 V AC** |
| **- isolation group** | | **C** |
| **- tested with** | | **1500 V AC** |
| **Current consumption** | | |
| **- from 5 V (internal)** | | **2 mA (+4 mA per active channel)** |
| **- from L+(for relay)** | | **16 mA per active channel** |
| **Power losses of the module** | **typically** | **6.5 W** |
| **Weight** | **approx.** | **0.8 kg (1.76 lb.)** |

## Digital Input/Output Module 40 x 24 V DC (with P Alarm) (6ES5 485-7LA11)




**Technical specifications**

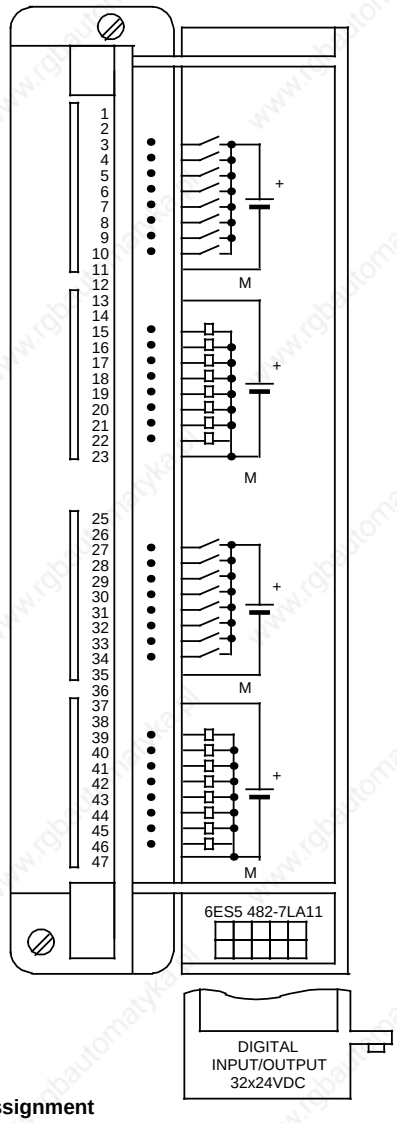
| | | |
|---|---|---|
| Number of inputs | | 24 |
| - in groups of | | 8 (of which 4 are alarm inputs) |
| (alarm function can be disabled) | | |
| Alarm signal | | via bus line PRAL_N |
| Galvanic isolation | | No |
| Input voltage | | |
| - Nominal value | | 24 V DC |
| - for "0" signal | | -30 to +5 V |
| - for "1" signal | | 13 to 30 V |
| Input current for "1" signal | typ. | 7.2 mA |
| Connection of 2-wire BERO | | Possible with quiescent input current 1.5 mA |
| Delay times | typ. | 3 ms |
| Delay times alarm inputs | typ. | 1.5 ms |
| Number of outputs | | 16 |
| - in groups of | | 8 |
| Galvanic isolation | | No |
| Load voltage L1+ (for inputs and upper output byte) | | |
| Load voltage L2+ (for lower output byte) | | |
| - Nominal value | | 24 V DC |
| - Permissible range | | 20 to 30 V |
| - Value at t < 0.5 s | | 35 V |
| Output voltage at "1" signal | min. | L+ -2.5 V |
| Voltage induced on circuit interruption limited to | | -15 V |
| Output current at "1" signal | | |
| - Nominal value | | 1.5 A, no overload protection (short-circuit protection activates at 3.6 A) |
| - Simultaneity factor in a group | | 0.5 |
| Output current/byte | max. | 6 A |
| - Lamp load | max. | 5 W |
| Residual current at "0" signal | max. | 1 mA |
| Parallel switching of outputs | | Not possible |
| Total permissible load | | 12 A |
| Short-circuit protection | | Electronic 3.6 A |
| Switching frequency | | |
| - with inductive load | max. | 0.5 Hz |
| - with resistive load | max. | 100 Hz |
| Cable length | | |
| - Shielded | max. | 1000 m/3281 ft |
| - Unshielded | max. | 600 m/1968.5 ft |
| Current consumption | | |
| - from 5 V (internal) | | < 100 mA |
| - from +L (without load) | | < 80 mA/output group |
| Power loss | typ. | 15 W |
| Weight | approx. | 0,7 kg |



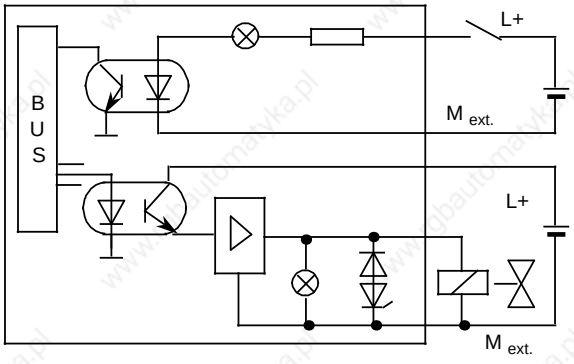

(also see Section 9.4)

## 15.2.6 Digital Input/Output Module

**Digital Input/Output Module 32 x 24 V DC; 0.5 A (6ES5 482-7LA11)**




**Terminal Assignment**




**Block diagram**

**Technical Specifications**

| | | |
|---|---|---|
| **Number of inputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **8** |
| **Input voltage** | | |
| **- rated value** | | **24 V DC** |

**The technical specifications for the inputs correspond to those of the 6ES5 430-7LA11 digital input module.**

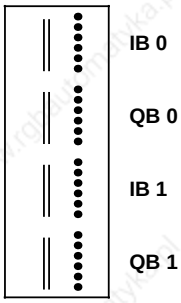
| | | |
|---|---|---|
| **Number of outputs** | | **16** |
| **Floating** | | **yes (optocoupler)** |
| **- in groups of** | | **8** |
| **Output current at "1" signal** | | |
| **- rated value** | | **0.5 A** |

**The technical specifications for the outputs correspond to those of the 6ES5 451-7LA12 digital output module.**

| | | |
|---|---|---|
| **Outputs** | **0 to 3 and 4 to 7<br>8 to 11 and 12 to 15** | **can be connected in parallel** |
| **Parallel current** | | **0.8 x $I_{rated}$** |
| **Permissible current of outputs** | | **100 % at 35 °C and 50 % at 55 °C (related to the sum of the currents of a group)** |
| **Current consumption** | | |
| **- from 5 V (internal)** | **maximum** | **50 mA** |
| **Power loss** | **typically** | **18 W** |
| **Weight** | **approx.** | **0.7 kg (1.54 lb.)** |

**The inputs and outputs are referenced under the same address (e.g. I 0.0 to I 1.7 and Q 0.0 to Q 1.7).**

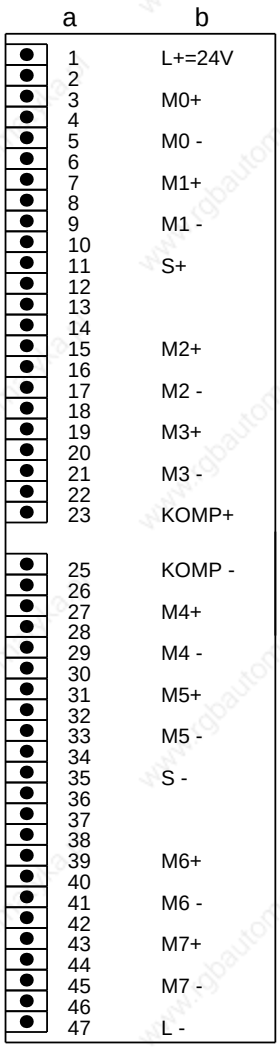
**Example: Addressing must be as follows for module start address "0":**



## 15.2.7 Analog Input Modules

### Analog Input Module 8 x I/V/PT 100, Floating (6ES5 460-7LA13)

**Terminal Assignment of the Front Connector**




a=Contact Pin No.
b=Assignment

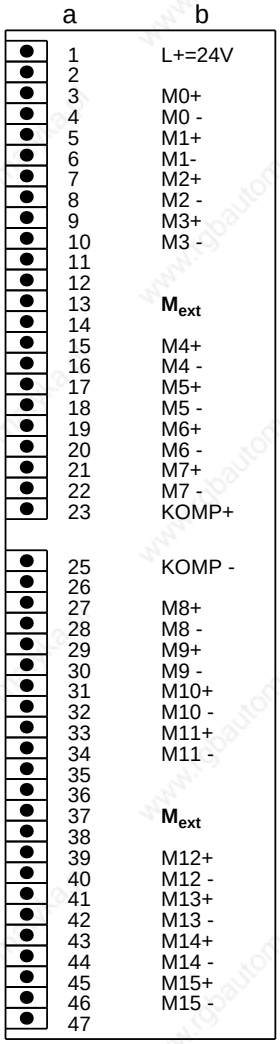
## Analog Input Module 8 x I/V/PT 100, Floating (6ES5 460-7LA13)

| Technical Specifications | | |
|---|---|---|
| Number of inputs | | 8 voltage/current inputs or 8 inputs for PT 100 |
| Floating | | yes (not for PT 100) |
| Input ranges (rated values) | | ±50 mV; ±500 mV; PT 100; ±1 V; ±5 V; ±10 V; ±20 mA; +4 to 20 mA (can be selected for four channels at a time using range cards) |
| Input resistance | | ±50 mV: 10 M<br>±500 mV: 10 M<br>PT 100: 10 M<br>±1 V: 90 k ; 2 ‰<br>±5 V: 50 k ; 2 ‰<br>±10 V: 50 k ; 2 ‰<br>±20 mA: 25 ; 1 ‰<br>±4 to 20 mA:31.25 ; 1 ‰ |
| Type of connection for sensors | | Two-wire connection; four-wire connection for PT 100 |
| Digital representation of the input signal | | 12 bits plus sign or 13 bits two's complement (2048 units=rated value) |
| Measuring principle | | integrating |
| Conversion principle | | voltage-time conversion (dual-slope) |
| Integration time (adjustable for optimum noise suppression) | | 20 msec. at 50 Hz<br>16.6 msec. at 60 Hz |
| Coding time (Single coding for 2048 units) | maximum | 60 msec. at 50 Hz<br>50 msec. at 60 Hz |
| Cycle time for<br>- 8 inputs | | 0.48 sec. at 50 Hz |
| Permissible voltage between inputs and between inputs and central grounding point (destruction limit) | maximum | 18 V or 75 V for max. 1 msec. and a duty cycle 1 : 20 |
| Permissible voltage between the reference potential of a nonfloating sensor and the central grounding point | maximum | 75 V DC /60 V AC |

| | | |
|---|---|---|
| Error indication for<br>- overranging | | yes (exceeding 4095 units) |
| - wire break of the sensor line | | can be designed for the ranges 50 mV, 500 mV and PT 100 (only measuring leads) |
| Wire-break test current (disconnectable) | | configurable |
| Noise suppression for f=n x (50/60 Hz±1 %) n=1, 2, ... | | |
| - common mode noise ($V_s$<1 V) | min. | 100 dB |
| - series mode noise (peak noise value < rated value of the range) | min. | 40 dB |
| Basic error limits | | ±50 mV : ±2 ‰<br>±500 mV : ±1.5 ‰<br>PT 100 : ±2 ‰<br>±1 V : ±3.5 ‰<br>±5 V : ±3.5 ‰<br>±10 V : ±3.5 ‰<br>±20 mA : ±2.5 ‰<br>+4 to 20 mA : ±2.5 ‰ |
| Operational error limits (0°C to 55°C) | | ±50 mV : ±5 ‰<br>±500 mV : ±4.5 ‰<br>PT 100 : ±5 ‰<br>±1 V : ±7.7 ‰<br>±5 V : ±7.7 ‰<br>±10 V : ±7.7 ‰<br>±20 mA : ±6.7 ‰<br>+4 to 20 mA : ±6.7 ‰ |
| Cable length<br>- shielded | maximum | 200 m (656 ft.); 50 m for± 50 mV |
| Front connector | | 46 pins |
| Isolation rating | | according to VDE 0160 |
| Rated isolation voltage (channel to channel)<br>- tested with | | 500 V |
| Rated isolation voltage (channel to ⏚ )<br>- tested with | | 500 V |
| Current consumption<br>- rated value<br>- ripple $V_{PP}$<br>- permissible range (including ripple) | | 24 V DC<br>3.6 V<br>20 to 30 V |
| Current consumption<br>- from 5 V (internal)<br>- from 24 V (external) | typically<br>typically | 0.15 A<br>0.1 A |
| Power losses of the module | typically | 3 W |
| Weight | approx. | 0.4 kg (0.88 lb.) |

## Analog Output Module 16 x I/V or 8 x PT 100, Nonfloating (6ES5 465-7LA13)

**Terminal Assignment of the Front Connector**




a=Contact Pin No.
b=Assignment

(Connection possibilities see Chapter 10)

## Analog Input Module 16 x I/V or 8 x PT 100, Nonfloating (6ES5 465-7LA13)

| Technical Specifications | | |
|---|---|---|
| Number of inputs | | 16 voltage/current inputs or 8 inputs for PT 100 |
| Type of inputs | | nonfloating |
| Input ranges (rated values) | | ±50 mV; ±500 mV; PT 100; ±1 V; ±5 V; ±10 V; ±20 mA; +4 to 20 mA (can be selected for four channels at a time using range cards) |
| Input resistance | | ±50 mV: 10 M<br>±500 mV: 10 M<br>PT 100: 10 M<br>±1 V: 90 k ; 2 ‰<br>±5 V: 50 k ; 2 ‰<br>±10 V: 50 k ; 2 ‰<br>±20 mA: 25 ; 1 ‰<br>±4 to 20 mA:31.25 ; 1‰ |
| Type of connection for sensors | | Two-wire connection; four-wire connection for PT 100 |
| Digital representation of the input signal | | 12 bit+sign or 13 bits two's complement (2048 units =rated value) |
| Measuring principle | | integrating |
| Conversion principle | | voltage-time conversion (dual-slope) |
| Integration time (adjustable for optimum noise suppression) | | 20 msec. at 50 Hz<br>16.6 msec. at 60 Hz |
| Coding time (single coding for 2048 units) | maximum | 60 msec. at 50 Hz<br>50 msec. at 60 Hz |
| Coding time<br>- 8 inputs<br>- 16 inputs | | <br>0.48 sec. at 50 Hz<br>0.96 sec. at 50 Hz |
| Permissible voltage between inputs and between inputs and central grounding point (destruction limit) | maximum | 18 V or 75 V for max. 1 msec. and a duty cycle of 1 : 20 |
| Permissible voltage between the reference potential of a nonfloating sensor and the central grounding point | maximum | ±1 V |
| Error indication for<br>- overranging<br>- wire break of the sensor line | | <br>yes (exceeding 4095 units)<br>Can be designed for the ranges 50 mV, 500 mV (PT 100) |
| Wire break test current disconnectable | | configurable |

| | | |
|---|---|---|
| Noise suppression for f=n x (50/60 Hz±1%) n=1, 2, ... | | |
| - common mode noise ($V_p$<1 V) | min. | 86 dB |
| - series mode noise (peak noise value < rated value of the range) | min. | 40 dB |

| Basic error limits | |
|---|---|
| ±50 mV | : ±2 ‰ |
| ±500 mV | : ±1.5 ‰ |
| PT 100 | : ±2‰ |
| ±1 V | : ±3.5 ‰ |
| ±5 V | : ±3.5 ‰ |
| ±10 V | : ±3.5 ‰ |
| ±20 mA | : ±2.5 ‰ |
| +4 to 20 mA | : ±2.5 ‰ |

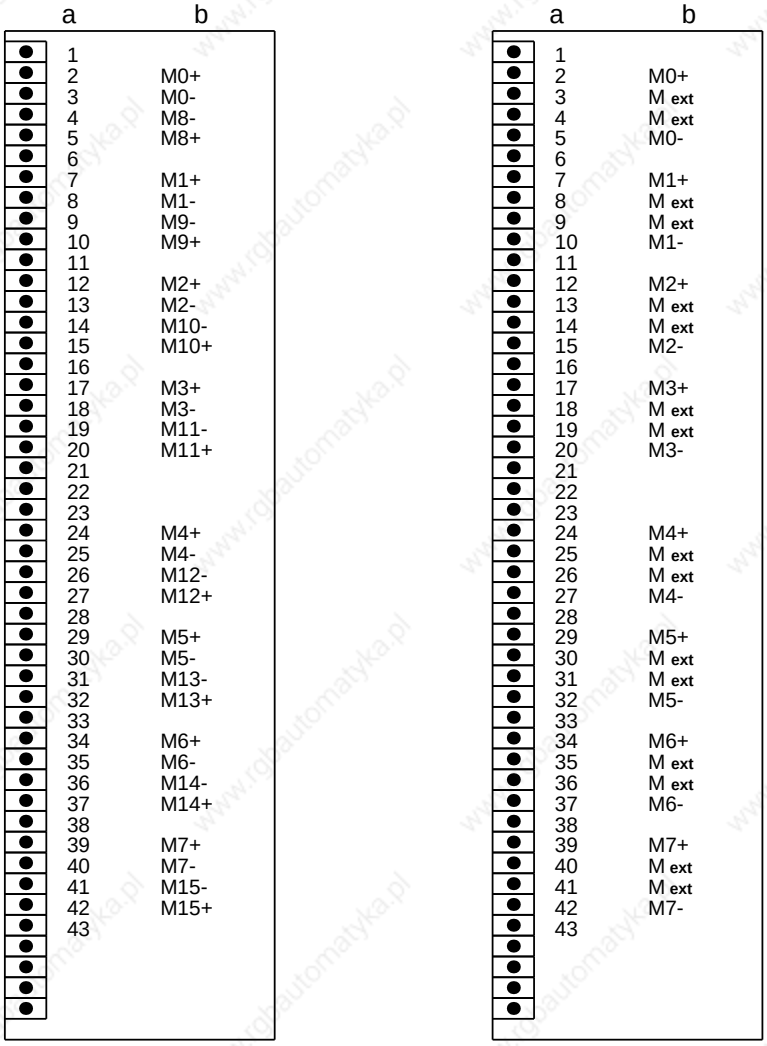
| Operational error limits (0°C to 55°C) | |
|---|---|
| ±50 mV | : ±5 ‰ |
| ±500 mV | : ±4,5 ‰ |
| PT 100 | : ±5 ‰ |
| ±1 V | : ±7.7 ‰ |
| ±5 V | : ±7.7 ‰ |
| ±10 V | : ±7.7 ‰ |
| ±20 mA | : ±6.7 ‰ |
| + to 20 mA | : ±6.7 ‰ |

| | | |
|---|---|---|
| Cable length<br>- shielded | maximum | 200 m;<br>50 m for±50 mV |
| Front connector | | 46 pin |
| Power supply<br>- rated value<br>- ripple $V_{PP}$<br>- permissible range (including ripple) | | <br>24 V DC [1]<br>3.6 V<br>20 to 30 V |
| Current consumption<br>- from 5 V (internal)<br>- from 24 V | <br>typically<br>maximum | <br>0.15 A<br>20 mA/transducer |
| Power losses of the module | typically | 0.75 W |
| Weight | approx. | 0.4 kg (0.88 lb.) |

[1] only required for two-wire transducers or for disconnecting the wire break test current

## Analog Input Module 16 x I/V or 8 x I/V, Floating (6ES5 466-3LA11)

**Terminal Assignment of the Front Connector**




Common-reference measurement

Differential measurement

a=Contact Pin No.
b=Assignment

(Connection possibilities see Chapter 10)

## Analog Input Module 16 x I/V or 8 x I/V, Floating (6ES5 466-3LA11)

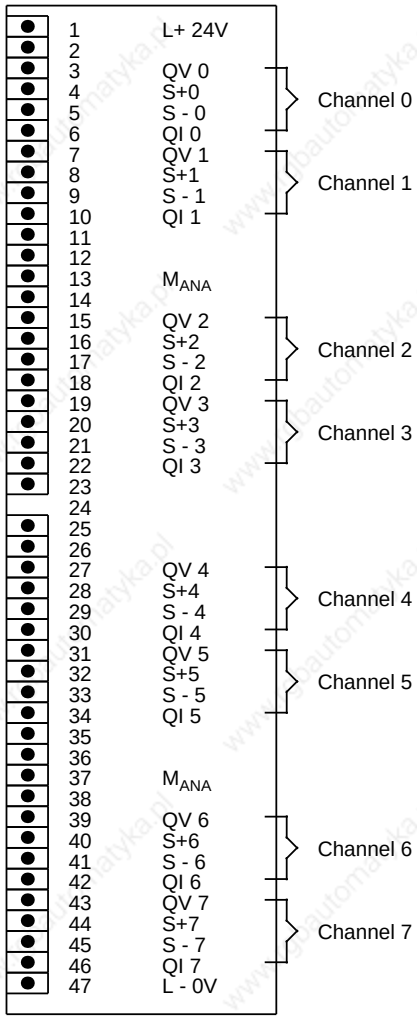
| Technical Specifications | |
|---|---|
| Number of inputs | 16 individual or 8 differential inputs in groups of 4 or 2 channels (switchable) voltage measurement or current measurement |
| Floating | yes |
| Input ranges | 0 to 20 mA, 4 to 20 mA, ±20 mA, 0 to 1.25 V, 0 to 2.5 V, 0 to 5 V, 1 to 5 V, 0 to 10 V, ±1.25 V, ±2.5 V, ±5 V, ±10 V |
| Input resistance | |
| - Voltage measuring range | 10 M |
| - Current measuring range | 125 |
| Type of connection for sensors | Two-wire connection |
| Digital representation of the input signal | Any of the following representations<br>- 12 bits two's complement<br>- 11 bits + sign<br>- 12 bits binary |
| Measuring principle | Momentary value decoding |
| Conversion principle | Successive approximation |
| Conversion time typically | 25 µsec. (per channel) |
| Coding time (per measured value) | 250 µsec. |
| Duration of cyclic sampling (scan time) | |
| - for 8 measured values | maximum 2 msec. |
| - for 8 measured values | maximum 4 msec. |
| Max. permissible input voltage (without destruction) | maximum ±30V (static) or ± 75V (Pulse for max. 1 msec. and a duty cycle 1:20) |
| Permissible isolation voltage between the reference potential and the central grounding point | maximum 60 V AC/75 V DC |
| Error indication for | |
| - Overflow | yes (overflow bit set) |
| - Internal errors | yes (error bit (= E bit) set) |
| Noise suppression common mode noise ($V_{PP}$=1 V) | minimum 70 dB |

| | |
|---|---|
| Basic error limits | |
| - Voltage ranges outside 0 to 1.25 V, +1.25 V | 0.1 % |
| - Current ranges and 0 to 1.25 V, +1.25 V | 0.12 % |
| Operational error limits (0 °C to 60 °C) | |
| - Voltage ranges outside 0 to 1.25 V, +1.25 V | 0.2 % |
| - Current ranges and 0 to 1.25 V, +1.25 V | 0.24 % |
| Individual errors | |
| - Linearity | 0.02 % |
| - Tolerance | 0.05 % |
| - Polarity error | 0.05 % |
| Temperature error | 0.005 %/K |
| Cable length | |
| - shielded | maximum 200 m (656 ft.) |
| Front connector | 43 pins |
| Isolation rating | to VDE 0160 |
| Rated isolation voltage (channels to grounding point) tested with | 500 V |
| Supply voltage | |
| - internal | +5 V+/- 5 % |
| - external | none |
| Internal current consumption | typically 0.7 A |
| Power losses of the module | typically 3.5 W |
| Weight | approx. 0.4 kg |
| Design | ES 902 |

## 15.2.8 Analog Output Modules

**Analog Output Modules 8 x± 10 V; 0 to 20 mA; Floating (6ES5 470-7LA12)**

**Terminal Assignment of the Front Connectors**




| | | |
|---|---|---|
| $M_{ANA}$ | = | common reference point of all current and voltage channels |
| QV x | = | voltage output channel x |
| QI x | = | current output channel x |
| S+x | = | sensor line+channel x |
| S - x | = | sensor line - channel x |

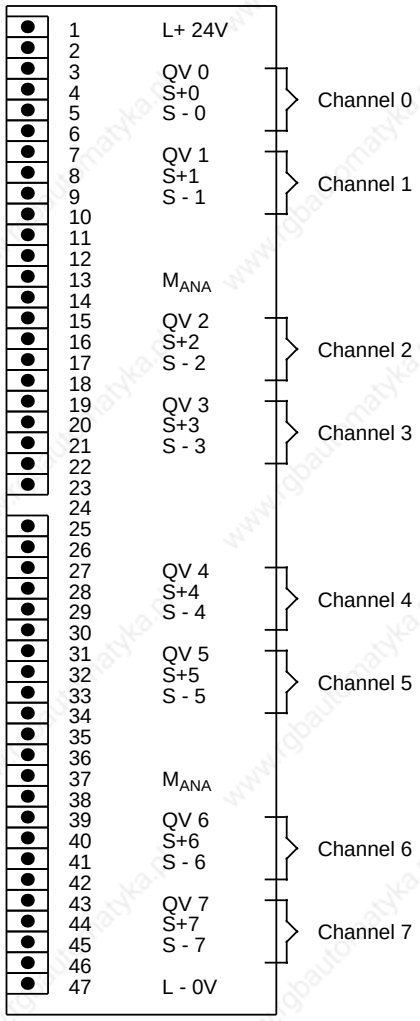
## Analog Output Module 8 x±10 V; 0 to 20 mA; Floating (6ES5 470-7LA12)

| Technical Specifications | | |
|---|---|---|
| Number of outputs | | 8 voltage and current outputs |
| Type of outputs | | floating (not between the inputs) |
| Output ranges (rated values) | | ± 10 V; 0 to 20 mA |
| Load resistance | | |
| - for voltage outputs | minimum | 3.3 k |
| - for current outputs | maximum | 300 |
| Load connection | | Load to $M_{ANA}$ - terminal |
| Digital representation of the output signal | | 11 bits plus sign (1024 units= rated value) |
| Conversion time | | 1 msec. |
| Permissible overload capability | approx. | 25 % (up to 1280 units ) |
| Short circuit protection | | yes |
| Short circuit current | approx. | 25 mA (for a voltage output) |
| Open circuit voltage | approx. | 18 V (for current output) |
| Voltage between the reference potential of the load ($M_{ANA}$ terminal) and the housing | maximum | 60 V AC/75 V DC |
| Linearity in the rated range | | ±2.5 ‰ ± 3 units |
| Operational error limits (0 °C to 55 °C) | | ±6 ‰ |
| Cable length | | |
| - shielded | maximum | 200 m (656 ft.) |
| Front connector | | 46 pin |
| Isolation rating | | according to VDE 0160 |
| Rated isolation voltage (outputs to ⏚ ) | | |
| - tested with | | 500 V |
| - max. capacitive load including cable capacity | | <100 nF |

| | | |
|---|---|---|
| Power supply | | |
| - rated value | | 24 V DC |
| - ripple $V_{PP}$ | | 3.6 V |
| - permissible range (ripple included) | | 20 to 30 V |
| Current consumption | | |
| - from 5 V (internal) | typically | 0.25 A |
| - from 24 V (external) | typically | 0.3 A |
| Power losses of the module | typically | 8.5 W |
| Weight | approx. | 0.4 kg (0.88 lb.) |

## Analog Output Module 8 x±10 V; Floating (6ES5 470-7LB12)

**Terminal Assignment of the Front Connector**




| | | |
|---|---|---|
| $M_{ANA}$ | = | common reference point of all current and voltage channels |
| QV x | = | voltage output channel x |
| S+x | = | sensor line + channel x |
| S - x | = | sensor line - channel x |

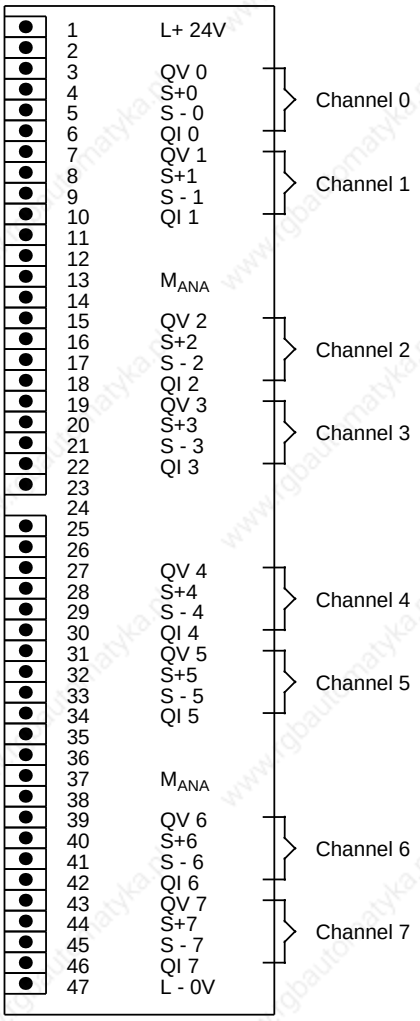
## Analog Ouptut Module 8 x ± 10 V; Floating (6ES5 470-7LB12)

| Technical Specifications | | |
|---|---|---|
| Number of outputs | | 8 voltage and current outputs |
| Type of outputs | | floating (not between the inputs) |
| Output ranges (rated value) | | ±10 V |
| Load resistance | minimum | 3.3 k |
| Load connection | | Load to $M_{ANA}$-terminal |
| Digital representation of the output signal | | 11 bits plus sign (1024 units= rated value) |
| Conversion time | | 1 msec. |
| Permissible overload capability | approx. | 25 % (up to 1280 units) |
| Short circuit protection | | yes |
| Short circuit current | approx. | 25 mA |
| Voltage between the reference potential of the load ($M_{ANA}$ terminal) and the housing | maximum | 60 V AC/75 V DC |
| Linearity in the rated range | | ±2,5 ‰ ±3 units |
| Operational error limits (0 °C to 55 °C) | | ±6 ‰ |
| Cable length - shielded | maximum | 200 m (656 ft.) |
| Front connector | | 46 pin |
| Isolation rating | | according to VDE 0160 |
| Rated isolation voltage (outputs to ⏚ ) - tested with | | 500 V |
| - max. capacitive load including cable capacity | | <100 nF |

| | | |
|---|---|---|
| Power supply | | |
| - rated value | | 24 V DC |
| - ripple $V_{PP}$ | | 3.6 V |
| - permissible range (ripple included) | | 20 to 30 V |
| Current consumption | | |
| - from 5 V (internal) | typically | 0.25 A |
| - from 24 V (external) | typically | 0.3 A |
| Power losses of the module | typically | 8.5 W |
| Weight | approx. | 0.4 kg (0.88 lb.) |

## Analog Output Module 8 x +1 to 5 V; +4 to 20 mA; Floating (6ES5 470-7LC12)

**Terminal Assignment of the Front Connector**




| | | |
|---|---|---|
| $M_{ANA}$ | = | common reference point of all current and voltage channels |
| QV x | = | voltage output channel x |
| QI x | = | current output channel x |
| S+x | = | sensor line + channel x |
| S - x | = | sensor line - channel x |

## Analog Output Module 8 x+1 to 5 V; +4 to 20 mA; Floating (6ES5 470-7LC12)

| Technical Specifications | | |
|---|---|---|
| Number of outputs | | 8 voltage and current outputs |
| Type of outputs | | floating (not between the inputs) |
| Output ranges (rated values) | | +1 to 5 V; +4 to 20 mA |
| Load resistance | | |
| - for voltage outputs | minimum | 3.3 k |
| - for current outputs | maximum | 300 |
| Load connection | | Load to $M_{ANA}$-terminal |
| Digital representation of the output signal | | 11 bits plus sign (1024 units=rated value) |
| Conversion time | | 1 msec. |
| Permissible overload | approx. | 25 % (up to 1280 units) |
| Short circuit protection | | yes |
| Short circuit current | approx. | 25 mA (for a voltage output) |
| Open circuit voltage | approx. | 18 V (for current output) |
| Voltage between the reference potential of the load ($M_{ANA}$ terminal) and the housing | maximum | 60 V AC/75 V DC |
| Linearity in the rated range | | ±2.5 ‰ ±3 units |
| Operational error limits (0 °C to 55 °C) | | ±6 ‰ |
| Cable length | | |
| - shielded | maximum | 200 m (656 ft.) |
| Front connector | | 46 pins |
| Isolation rating | | according to VDE 0160 |
| Rated isolation voltage (outputs to ⏚ ) | | |
| - tested with | | 500 V |
| - max. capacitive load including cable capacity | | <100 nF |

| | | |
|---|---|---|
| Power supply | | |
| - rated value | | 24 V DC |
| - ripple $V_{PP}$ | | 3.6 V |
| - permissible range (ripple included) | | 20 to 30 V |
| Current consumption | | |
| - from 5 V (internal) | typically | 0.25 A |
| - from 24 V (external) | typically | 0.3 A |
| Power losses of the module | typically | 8.5 W |
| Weight | approx. | 0.4 kg (0.88 lb.) |

### 15.2.9 Intelligent Input/Output Modules

Table 15-1 lists the intelligent input/output modules you can use with the S5-115U programmable controller.

**Table 15-1. Overview of Intelligent Input/Output Modules**

| Intelligent Input/Output Modules* | Current Consumption (int. at 5 V) | Fan required? | Adapter casing required? |
|---|---|---|---|
| IP 240<br>counter and position decoder | 0.6** A | no | yes |
| IP 241<br>digital position decoder | 1 A | yes | yes |
| IP 242<br>counter module | 0.9 A | no | yes |
| IP 243<br>analog module | 0.6 A | no | yes |
| IP 244<br>temperature control module | 0.8 A | no | yes |
| IP 245<br>valve control module | 0.2 A | yes | yes |
| IP 246<br>positioning module | 1.0 A | no | yes |
| IP 247-4UA11<br>IP 247-4UA21<br>positioning module | 0.8 A | yes<br>no | yes |
| IP 252<br>closed-loop control module | 2.3 A | no | yes |
| WF 625<br>positioning module | 1.6 A | yes | yes |
| IP260<br>closed-loop control module | 1A*** | no | yes |
| IP 261<br>proportioning module | 0.05A | no | yes |

* please see the catalog for the order numbers for the modules or the manuals

** without sensor power supply

*** current consumption external with 24 V, without load

## 15.2.10 Communications Processors

Table 15-2 lists the communications processors that you can use with the S5-115U programmable controller.

**Table 15-2. Overview of Communications Processors**

| Communications Processors* | Current Consumption (int. at 5V) | Fan required? | Adapter casing required? |
|---|---|---|---|
| CP 513 (bubble memory)<br>- 128 x $2^{10}$ byte<br>- 256 x $2^{10}$ byte | 2.3 A | yes | yes |
| CP 524<br>Computer link | 1.5 A | yes | yes |
| CP 525<br>Listing/Computer link | 1.8 A | yes | yes |
| CP 526<br>Listing/Computer link | 2.2 A | yes | yes |
| CP 530A<br>Configuring a SINEC-L1-local area network | 1.0 A | yes | yes |
| CP 530<br>Configuring a SINEC-L1-local area network | 1.0 A | no | no |
| CP 5430<br>Configuring a SINEC-L2-local area network | 0.33 A | no** | yes |
| CP 143-0AB..<br>Configuring a SINEC-H1-local area network<br>CP 535 | 2.5 A<br><br>4 A | yes | yes |
| CP 523<br>Serial input/output | 0.13 A | no | yes |
| CP 527<br>- for monochrome CRT units<br>- for colour CRT units | 1.5 A | yes | yes |
| CP 551<br>Hard disk memory | 4.5 A | yes | yes*** |
| CP 552-1<br>CP 552-2<br>Diagnostics processor | 1.8 A<br>3.2 A | no<br>no | yes |

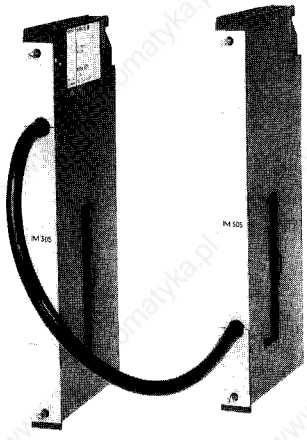
* please see the catalog for the order numbers for the modules or the manuals

** see Section 3 "Installation Guidelines"

*** can only be plugged in with adapter casing 6ES5 491-0LC11

## 15.2.11 Interface Modules

### IM 305 Interface Module (6ES5 305-7LA11)



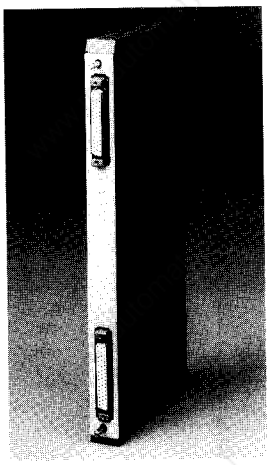

R-ST 52-069

**Technical Specifications**

| | | |
|---|---|---|
| Current supplied to the EU | maximum | 1 A |
| Current consumption (5 V; own consumption) | | 10 mA |
| Cable length | | 0.5 m (1.6 ft.) |
| Weight (total) | approx. | 0.6 kg (1.3 lb.) |

The IM 305 interface module is used for central connection of one expansion unit (EU) to a central controller (CC) (see also Chapter 3). It consists of two modules that are permanently connected to each other by a cable (0.5 m, or 1.6 ft.).

### IM 306 Interface Module (6ES5 306-7LA11)



R-ST 52-052

**Technical Specifications**

| | | |
|---|---|---|
| Current supplied to the EU | maximum | 2 A |
| Current consumption (5 V; own consumption) | | 50 mA |
| Weight | approx. | 0.6 kg (1.3 lb.) |

**Accessories**

| | |
|---|---|
| 705 connecting cable (see Catalog ST 52.3) | 6ES5 705-0AF00<br>6ES5 705-0BB50 |

The IM 306 interface module is used for central connection of up to three expansion units (EUs) to a central controller (CC) (see also Chapter 3).

## IM 304 Interface Module

**(6ES5 304-3UA11)**

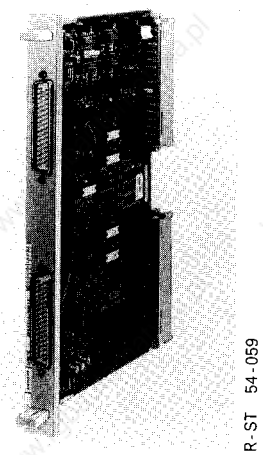
**Technical Specifications**

| | |
|---|---|
| Current consumption (at 5 V) | maximum 1.5 A |
| Weight | approx. 0.3 kg (0.7 lb.) |

The IM 304 interface module is used in combination with the IM 314 interface module for distributed connection (up to 600 m, or 1969 ft.) of expansion units (EUs) to a central controller (CC) (see also Chapter 3).

## IM 314 Interface Module

**(6ES5 314-3UA11)**




**Technical Specifications**

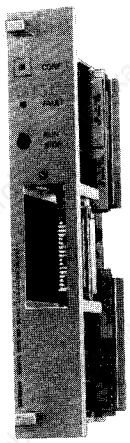
| | |
|---|---|
| Current consumption (at 5 V) | maximum 1.0 A |
| Weight | approx. 0.3 kg (0.7 lb.) |

**Accessories**

| | |
|---|---|
| Adapter casing | 6ES5 491-0LA12 |
| Termination connector for AS 314 | 6ES5 760-1AA11 |
| 721 connecting cable (see Catalog ST 52.3) | |

The IM 314 interface module is used in combination with the IM 304 interface module for distributed connection (up to 600 m, or 1969 ft.) of expansion units (EUs) to a central controller (CC) (see also Chapter 3).

## IM 307 Expansion Unit Interface Module (6ES5 307-3UA11)

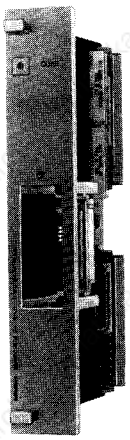


EWA 4098/11

**Technical Specifications**

| | |
|---|---|
| Current consumption (at 5 V) | 1 A |
| Weight | 0.4 kg (0.9 lb.) |

## IM 317 Central Controller Interface Module (6ES5 317-3UA11)



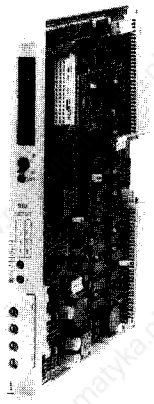
EWA 4098/16

**Technical Specifications**

| | |
|---|---|
| Current consumption (at 5 V) | 1 A |
| Weight | 0.4 kg (0.9 lb.) |

## IM 308 Interface Module

**(6ES5 308-3UA12)**



R-ST 54-076

**Technical Specifications**

| | |
|---|---|
| Current consumption (at 5 V) | maximum 0.5 A |
| Weight | approx. 0.4 kg (0.9 lb.) |

The IM 308 interface module is used to connect the ET 100 electronic terminator to the S5-115U programmable controller.

## IM 318 Interface Module

**(6ES5 318-3UA11)**

GWA 3961/7

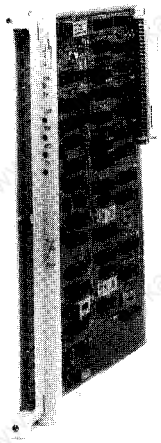
**Technical Specifications**

| | |
|---|---|
| Current consumption (at 5 V) | maximum 0.3 A |
| Weight | approx. 0.34 kg (0.75 lb.) |

The IM 318 interface module is used in conjunction with the IM 308 interface module for the distributed configuration (up to 3000 m or 10,000 ft.) of expansion units and one central controller.

## 15.2.12 The 313 Watchdog Module

**313 Watchdog Module** **(6ES5 313-3AA12)**



**Technical Specifications**

| | | |
|---|---|---|
| Current consumption (at 5 V) | maximum | 0.4 A |

For further technical specifications see Manual C 79 000-B85000-C266-1

**Accessories**

| | |
|---|---|
| Adapter casing | 6ES5 491-0LA12 |

The 313 Watchdog module monitors signals on the S5 bus. It can be used to check the connection between a central controller and an expansion unit for open circuits and short circuits.

## 15.3 Accessories

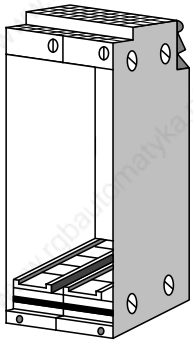
**Adapter Casing for Two Printed Circuit Boards (6ES5 491-0LB12)**

**Technical Specifications**

| | |
|---|---|
| Dimensions (w×h×d) in mm | 43 x 303 x 187 |
| Weight | approx. 0.9 kg (2 lb.) |

Even modules which are not of the block type can be used in the S5-115U, provided an adapter casing is available.
The adapter casing can take one module or, in the case of the CR 700-3 subrack, two modules, but only one double-width module IP241, IP 245, IP 246 and IP 247 (self-ventilated model) IP 252 and CP 535.

**Adapter Casing for CP 551 Bulk Storage Memory or for up to 6 Printed Circuit Boards (6ES5 491-0LC11)**



**Technical Specifications**

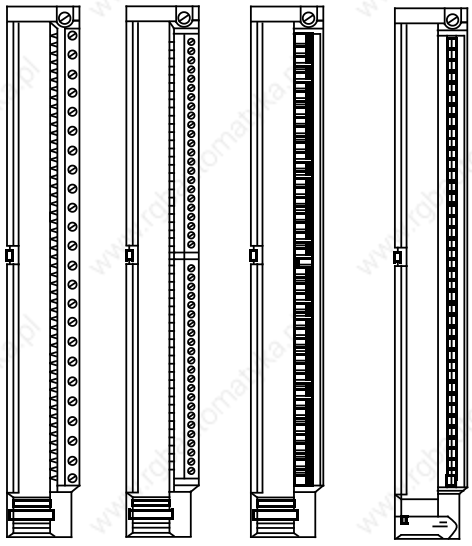
| | |
|---|---|
| Dimensions (w×h×d) in mm | 86 x 303 x 187 |
| Weight | approx. 0,8 kg (1.8 lb.) |

Even modules which are not of the block type can be used in the S5-115U, provided an adapter casing is available.
The adapter casing can take two double-width or 4 single-width modules; the SIMATIC S5 CP 580/581 can be used in the S5-115U programmable controller with the help of this adapter casing.

**490 Front Connector**

Screw terminals | Crimp-snap-in | Spring-loaded connectors

24-pole | 46-pole | 46-pole | 46-pole



**Technical Specifications**
see Catalog ST 52.3

**490 Front Connector**

- for screw-type terminals
  - 24-pole **6ES5 490-7LB11**
  - 46-pole **6ES5 490-7LB21**

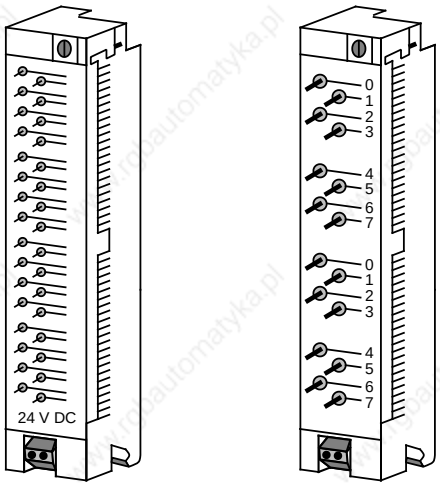
763 jumper comb 6ES5 763-7LA11
(for use with screw-terminal front connectors)

- for crimp snap-in connections 46-pole
  - without crimp contacts **6ES5 490-7LA21**
  - with 50 crimp contacts **6ES5 490-7LA11**

Crimp contacts (250) 6XX3 070
Crimping tool for crimping the crimp contacts 6XX3 071
Extraction tool for crimp contacts 6ES5 497-8MA11

- for spring-loaded connectors
  - 46-pole **6ES5 490-7LC11**

**Simulator**




**Technical Specifications**
see Catalog ST 52.3

**Simulator**

- 32 switches/buttons 24 V DC **6ES5 490-7SA11**
  can be plugged into 6ES5 420-7LA11
  6ES5 430-7LA11

- 16 switches/buttons 24/48/60 V AC/DC **6ES5 490-7SA21**
  can be plugged into 6ES5 431-7LA11
  6ES5 432-7LA11
  6ES5 435-7LA11
  6ES5 435-7LB12
  6ES5 436-7LA11
  6ES5 436-7LB12

**Fan Subassembly**

If the 6ES5 951-7LD21/51 or 6ES5 951-7ND41 power supply modules carry a load of more than 7 A, or if modules with a high power consumption are used, a fan subassembly is necessary.

**Technical Specifications (6ES5 981-0HA11 and 6ES5 981-0HB11)**

| Fan | 6ES5 981-0HA11 | 6ES5 981-0HB11 |
|---|---|---|
| Input voltage | | |
| - rated value | 230/115 V AC | 230/115 V AC |
| - tolerance | - 10 % to+10 % | - 10 % to+10 % |
| Network frequency | | |
| - Rated value | 50/60 Hz | 50/60 Hz |
| Input current | typically 420 mA | typically 420 mA |
| Contact rating | | |
| - with ohmic load | 5.0 A at 230 V AC | 5.0 A at 230 V AC |
| 2.5 A | at 30 V DC | 2.5 A at 30 V DC |
| - with inductive load | 1.5 A at 230 V AC | 1.5 A at 230 V AC |
| 0.5 A | at 30 V DC | 0.5 A at 30 V DC |
| - Life span | | |
| Operating cycles | $1.5 \cdot 10^6$ AC11 | $1.5 \cdot 10^6$ AC11 |
| Degree of protection | IP20 to DIN 40 050 | IP20 to DIN 40 050 |
| Radio interference suppression level | A to VDE 0871 | A to VDE 0871 |
| Dimensions | 423 x 110 x 135 | 294 x 110 x 135 |
| wxhxd (mm) | 16.6 x 3.93 x 5.31 in. | 11.5 x 3.93 x 5.31 in. |
| Weight | 1.5 kg (3.3 lb.) | 1.4 kg (3 lb.) |

**Accessories**

| | | |
|---|---|---|
| Installation parts | 6ES5 981-0GA11 | 6ES5 981-0GB11 |
| Filter mat unit | 6ES5 981-0JA11 | 6ES5 981-0JB11 |

**Fan Subassembly (Continued)**

**Technical Specifications (6ES5 981-0HA21 and 6ES5 981-0HB21)**

| Fan | 6ES5 981-0HA21 | 6ES5 981-0HB21 |
|---|---|---|
| Input voltage | | |
| - rated value | 24 V DC | 24 V DC |
| - permissible range (including ripple) | +20 V to+30 V | +20 V to+30 V |
| Input current | typically 800 mA | typically 800 mA |
| Contact rating | | |
| - with ohmic load | 5.0 A at 230 V AC<br>2.5 A at 30 V DC | 5.0 A at 230 V AC<br>2.5 A at 30 V DC |
| - with inductive load | 1.5 A at 230 V AC<br>0.5 A at 30 V DC | 1.5 A at 230 V AC<br>0.5 A at 30 V DC |
| - Life span | | |
| Operating cycles | 1.5·106 DC11 | 1.5·106 DC11 |
| Degree of protection | IP20 to DIN 40 050 | IP20 to DIN 40 050 |
| Radio interference suppression level | A to VDE 0871 | A to VDE 0871 |
| Dimensions | | |
| wxhxd (mm) | 423 x 110 x 135<br>16.6 x 3.93 x 5.31 in. | 294 x 110 x 135<br>11.5 x 3.93 x 5.31 in. |
| Weight | 1.5 kg (3.3 lb.) | 1.4 kg (3 lb.) |

**Accessories**

| | | |
|---|---|---|
| Installation parts | 6ES5 981-0GA11 | 6ES5 981-0GB11 |
| Filter mat unit | 6ES5 981-0JA11 | 6ES5 981-0JB11 |

**Back-Up Battery** **(6EW1 000-7AA)**

**Technical Specifications**

| | | |
|---|---|---|
| Lithium battery (3.4 V/5.2 Ah) | | |
| - back-up time (at 25 °C and constant backup of the CPU with memory submodule) | approx. | 2 years |
| - service life (at 25 °C) | approx. | 5 years |
| - external battery backup | | 3.4 to 9 V |

**Types of Fuses**

| | | |
|---|---|---|
| **Wickmann 19231** | **2.5 A FF** | **6ES5 980-3BC21** |
| | **4 A FF** | **6ES5 980-3BC51** |
| | **10 A FF** | **6ES5 980-3BC41** |
| **Gould GAB4** | | |
| **Bussmann ABC4** | | |

**Appendices**


Appendix A . . . . . Operations List
Appendix B . . . . . Maintenance
Appendix C . . . . . Module Slots
Appendix D . . . . . Active and Passive Faults in Automation Equipment
Appendix E . . . . . SIEMENS Addresses Worldwide

## A Operations List


A.1 Explanation of the Operations List . . . . . A - 1
A.2 Basic Operations . . . . . 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 Operations List

## A.1 Explanation of the Operations List

| Abbreviation | Explanation |
|---|---|
| ACCUM 1 | Accumulator 1 (When accumulator 1 is loaded, any existing contents are shifted into accumulator 2.) |
| ACCUM 2 | Accumulator 2 |
| CC0/CC1 | Condition code 0/Condition code 1 |
| CSF | STEP 5 control system flowchart method of representation |
| Formal operand | Expression with a maximum of 4 characters. The first character must be a letter of the alphabet. |
| LAD | STEP 5 ladder diagram method of representation |
| OV | Overflow. This condition code bit is set if, e.g., a numerical range is exceeded during arithmetic operations. |
| PII | Process image input |
| PIQ | Process image output |
| RLO | Result of logic operation |
| RLO affected? Y/N | The RLO is affected/not affected by the operation. |
| RLO dependent? Y<br>Y /<br>N | The statement is executed only if the RLO is "1".<br>The statement is executed only on positive/negative edge change of the RLO.<br>The statement is always executed. |
| RLO reloaded? Y<br>N | The RLO does not change. The RLO cannot be combined any further. When the next binary operation takes place (but not assignment operation), the RLO is reloaded.<br>Depending on whether the operation affects the RLO, the RLO is combined further or left unchanged according to the operation and the status of the bit that was scanned. |
| STL | STEP 5 statement list method of representation |

| Abb. | Explanation | Permissible operand value range for | | | |
|---|---|---|---|---|---|
| | | **CPU 941** | **CPU 942** | **CPU 943** | **CPU 944** |
| BN | Byte constant<br>(fixed-point number) | - 128 to + 127 | | | |
| C | Counter<br>- for the bit test and set operations (system operations) | 0 to 127<br>0.0 to 127.15 | | | |
| D | Data word (1 bit)<br>- for load operations (supplementary operations) and transfer operations (system operations)<br>- for bit test and set operations (system operations) | 0.0 to 255.15 | | | |
| DB | Data block | 2 to 255 | | | |
| DL | Data word (left byte) | 0 to 255 | | | |
| DR | Data word (right byte) | 0 to 255 | | | |
| DW | Data word | 0 to 255 | | | |
| F | Flag | 0.0 to 255.7 | | | |
| FB | Function block | 0 to 255 | | | |
| FW | Flag word | 0 to 254 | | | |
| FY | Flag byte | 0 to 255 | | | |
| I | Input | 0.0 to 127.7 | | | |
| IB | Input byte | 0 to 127 | | | |
| IW | Input word | 0 to 126 | | | |
| 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 two alphanumeric characters | | | |
| KT | Constant (time) | 0.0 to 999.3 | | | |
| KY | Constant (2 bytes) | 0 to 255 (per byte) | | | |
| OB [1] | Organization block | 0 to 255 | | | |

| Abb. | Explanation | Permissible operand value range for | | | |
|---|---|---|---|---|---|
| | | CPU 941 | CPU 942 | CPU 943 | CPU 944 |
| PB | Program block<br>(with block call and return operations) | 0 to 255 | | | |
| PB | Peripheral byte<br>- Digital inputs<br>- Analog inputs<br>- Digital outputs<br>- Analog outputs | <br>0 to 127<br>128 to 255<br>0 to 127<br>128 to 255 | | | |
| PW | Peripheral word<br>- Digital inputs<br>- Analog inputs<br>- Digital outputs<br>- Analog outputs | <br>0 to 126<br>128 to 254<br>0 to 126<br>128 to 254 | | | |
| Q | Output | 0 to 127.7 | | | |
| QB | Output byte | 0 to 127 | | | |
| QW | Output word | 0 to 126 | | | |
| RS | System data range<br>- for load operations (supplementary operations) and transfer operations (system operations)<br>- for bit test and set operations (system operations) | <br>0 to 255<br>0.0 to 255.15 | | | |
| SB | Sequence block | 0 to 255 | | | |
| T | Timer<br>- for the bit test and set operations (system operations) | 0 to 127<br>0.0 to 127.15 | | | |

1 See Section 7.3.1 for an overview of the organization blocks and their functions

## Note

Please note that the execution times specified in A.2 to A.4 are guideline values. This is determined by the processor architecture. The operation runs in the standard processor or in the STEP 5 coprocessor, depending on CPU type.
The strict execution time is increased by a transfer time when changing from direct processing in the coprocessor to interpretive processing in the standard processor. These transfer times are contained in the execution times specified, based on an operation mix.

## A.2 Basic 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? | | | Typical execution time in µs | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Boolean Logic Operations** | | | | | | | | | | | | | |
| A | • | • | • | • | • | N | Y | N | 1.6 | 1.6 | 0.8 | 0.8 | Scan operand for "1" and combine with RLO through logic AND. |
| AN | • | • | • | • | • | N | Y | N | 1.6 | 1.6 | 0.8 | 0.8 | Scan operand for "0" and combine with RLO through logic AND. |
| O | • | • | • | • | • | N | Y | N | 1.6 | 1.6 | 0.8 | 0.8 | Scan operand for "1" and combine with RLO through logic OR. |
| ON | • | • | • | • | • | N | Y | N | 1.6 | 1.6 | 0.8 | 0.8 | Scan operand for "0" and combine with RLO through logic OR. |
| O | | | | | | N | Y | Y | 1.6 | 1.6 | 0.8 | 0.8 | Combine AND operations through logic OR. |
| A( | | | | | | N | Y | Y | 1.6 | 1.6 | 0.8 | 0.8 | Combine expressions enclosed in parentheses through logic AND (6 levels). |
| O( | | | | | | N | Y | Y | 1.6 | 1.6 | 0.8 | 0.8 | Combine expressions enclosed in parentheses through logic OR (6 levels). |
| ) | | | | | | N | Y | N | 1.6 | 1.6 | 0.8 | 0.8 | Close parenthesis (conclusion of a parenthetical expression). |
| **Set/Reset Operations** | | | | | | | | | | | | | |
| S | • | • | • | | | Y | N | Y | 1.6 | 1.6 | 0.8 | 0.8 | Set operand to "1". |
| R | • | • | • | | | Y | N | Y | 1.6 | 1.6 | 0.8 | 0.8 | Reset operand to "0". |
| = | • | • | • | | | N | N | Y | 1.6 | 1.6 | 0.8 | 0.8 | Assign value of RLO to operand. |
| **Load Operations** | | | | | | | | | | | | | |
| L | IB | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load an input byte from the PII into ACCUM 1. |
| L | QB | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load an output byte from the PIQ into ACCUM 1. |
| L | IW | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load an input word from the PII into ACCUM 1:<br>byte n ACCUM 1 (bits 8 - 15);<br>byte n+1 ACCUM 1 (bits 0 - 7). |

☒ 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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Load Operations (Continued)** | | | | | | | | | | | | | |
| L | QW | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load an output word from the PIQ into ACCUM 1: byte n ACCUM 1 (bits 8 - 15); byte n+1 ACCUM 1 (bits 0 - 7). |
| L | PB | | | | | N | N | N | 93* | 93* | 93* | 4* | Load an input byte from the digital/analog input modules into ACCUM 1. |
| L | PW | | | | | N | N | N | 107* | 107* | 107* | 4.8** | Load a peripheral word from the digital/analog inputs into ACCUM 1: byte n ACCUM 1 (bits 8 - 15); byte n+1 ACCUM 1 (bits 0 - 7). |
| L | FY | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load a flag byte into ACCUM 1. |
| L | FW | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load a flag word into ACCUM 1: byte n ACCUM 1 (bits 8 - 15); byte n+1 ACCUM 1 (bits 0 - 7). |
| L | DL | | | | | N | N | N | 3.4 | 3.4 | 1.7 | 1.7 | Load a data word (left-hand byte) of the current data block into ACCUM 1. |
| L | DR | | | | | N | N | N | 3.4 | 3.4 | 1.7 | 1.7 | Load a data word (right-hand byte) of the current data block into ACCUM 1. |
| L | DW | | | | | N | N | N | 3.9 | 3.9 | 2 | 2 | Load a data word of the current data block into ACCUM 1: byte n ACCUM 1 (bits 8 - 15); byte n+1 ACCUM 1 (bits 0 - 7). |
| L | KB | | | | | N | N | N | 2.8 | 2.8 | 1.4 | 1.4 | Load a constant (1-byte number) into ACCUM 1. |
| L | KS | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load a constant (2 characters in ASCII format) into ACCUM 1. |
| L | KF | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load a constant (fixed-point number) into ACCUM 1. |
| L | KH | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load a constant (hexadecimal code) into ACCUM 1. |
| L | KM | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load a constant (bit pattern) into ACCUM 1. |
| L | KY | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load a constant (bit pattern) into ACCUM 1. |
| L | KT | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load a constant (count in BCD) into ACCUM 1. |

* + ready delay time of the referenced I/O modules (digital I/O: 2 µs/byte, analog I/O: 16 µs/byte)
** + 2 x ready delay time of the referenced I/O modules

☒ 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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Load Operations (cont.)** | | | | | | | | | | | | | |
| L | KC | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load a constant (count in BCD) into ACCUM 1. |
| L | | | | • | • | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Load a time or count (in binary code) into ACCUM 1. |
| LD | | | | • | • | N | N | N | 3.5 | 3.5 | 1.8 | 1.8 | Load times or counts (in BCD) into ACCUM 1. |
| **Transfer Operations** | | | | | | | | | | | | | |
| T | IB | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Transfer the contents of ACCUM 1 to an input byte (into the PII). |
| T | QB | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Transfer the contents of ACCUM 1 to an output byte (into the PIQ). |
| T | IW | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Transfer the contents of ACCUM 1 to an input word (into the PII): ACCUM 1 (bits 8 - 15) byte n; ACCUM 1 (bits 0 - 7) byte n+1. |
| T | QW | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Transfer the contents of ACCUM 1 to an output word (into the PIQ): ACCUM 1 (bits 8 - 15) byte n; ACCUM 1 (bits 0 - 7) byte n+1. |
| T | PB | | | | | N | N | N | 67* | 67* | 67* | 3.9* | Transfer the contents of ACCUM 1 to an I/O byte of the digital output modules with updating of the PIQ or analog output modules. |
| T | PW | | | | | N | N | N | 85* | 85* | 85* | 4.7** | Transfer the contents of ACCUM 1 to an I/O byte of the digital output modules with updating of the PIQ or the analog output modules. |
| T | FY | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Transfer the contents of ACCUM 1 to a flag byte. |
| T | FW | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Transfer the contents of ACCUM 1 to a flag word (into the PIQ): ACCUM 1 (bits 8 - 15) byte n; ACCUM 1 (bits 0 - 7) byte n+1. |
| T | DL | | | | | N | N | N | 2.2 | 2.2 | 1.1 | 1.1 | Transfer the contents of ACCUM 1 to a data word (left-hand byte). |
| T | DR | | | | | N | N | N | 2.2 | 2.2 | 1.1 | 1.1 | Transfer the contents of ACCUM 1 to a data word (right-hand byte). |
| T | DW | | | | | N | N | N | 2.7 | 2.7 | 1.4 | 1.4 | Transfer the contents of ACCUM 1 to a data word. |

* + ready delay time of the referenced I/O modules (digital I/O: 2 µs/byte, analog I/O: 16 µs/byte)
** + 2 x ready delay time of the referenced I/O modules

☒ 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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Timer Operations** | | | | | | | | | | | | | |
| SP | | | | • | | Y | N | Y | 3.7 | 3.7 | 1.9 | 1.9 | Start a timer (stored in ACCUM 1) as signal-contracting pulse on the leading edge of the RLO |
| SE | | | | • | | Y | N | Y | 3.7 | 3.7 | 1.9 | 1.9 | Start a timer (stored in ACCUM 1) as extended pulse (signal contracting and stretching) on the leading edge of the RLO. |
| SR | | | | • | | Y | N | Y | 3.7 | 3.7 | 1.9 | 1.9 | Start an on-delay timer (stored in ACCUM 1) on the leading edge of the RLO. |
| SS | | | | • | | Y | N | Y | 3.7 | 3.7 | 1.9 | 1.9 | Start a stored on-delay timer (stored in ACCUM 1) on the leading edge of the RLO. |
| SF | | | | • | | Y | N | Y | 3.7 | 3.7 | 1.9 | 1.9 | Start an off-delay timer (stored in ACCUM 1) on the trailing edge of the RLO. |
| R | | | | • | | Y | N | Y | 3.7 | 3.7 | 1.9 | 1.9 | Reset a timer if RLO="1". |
| **Counter Operations** | | | | | | | | | | | | | |
| CU | | | | | • | Y | N | Y | 3.7 | 3.7 | 1.9 | 1.9 | Counter counts up 1 on the leading edge of the RLO. |
| CD | | | | | • | Y | N | Y | 3.7 | 3.7 | 1.9 | 1.9 | Counter counts down 1 on leading edge of the RLO. |
| S | | | | | • | Y | N | Y | 3.7 | 3.7 | 1.9 | 1.9 | Set counter if RLO="1". |
| R | | | | | • | Y | N | Y | 3.7 | 3.7 | 1.9 | 1.9 | Reset counter if RLO="1". |
| **Arithmetic Operations** | | | | | | | | | | | | | |
| +F | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Add two fixed-point numbers: ACCUM 1 + ACCUM 2. Result evaluation via ANZ 1/ANZ 0/0V |
| - F | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Subtract two fixed-point numbers: ACCUM 1 - ACCUM 2. Result evaluation via ANZ 1/ANZ 0/0V |

☒ 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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Comparison Operations** | | | | | | | | | | | | | |
| !=F | | | | | | N | Y | N | 1.6 | 1.6 | 0.8 | 0.8 | Compare two fixed-point numbers for "equal to". If ACCUM 2=ACCUM 1, the RLO is "1". CC 1/CC 0 are affected. |
| ><F | | | | | | N | Y | N | 1.6 | 1.6 | 0.8 | 0.8 | Compare two fixed-point numbers for "not equal to". If ACCUM 2 ACCUM 1, the RLO is "1". CC 1/CC 0 are affected. |
| >F | | | | | | N | Y | N | 1.6 | 1.6 | 0.8 | 0.8 | Compare two fixed-point numbers for "greater than". If ACCUM 2>ACCUM 1, the RLO is "1". CC 1/CC 0 are affected. |
| >=F | | | | | | N | Y | N | 1.6 | 1.6 | 0.8 | 0.8 | Compare two fixed-point numbers for "greater than or equal to". If ACCUM 2 ACCUM 1, the RLO is "1". CC 1/CC 0 are affected. |
| <F | | | | | | N | Y | N | 1.6 | 1.6 | 0.8 | 0.8 | Compare two fixed-point numbers for "less than". If ACCUM 2<ACCUM 1, the RLO is "1". CC 1/CC 0 are affected. |
| <=F | | | | | | N | Y | N | 1.6 | 1.6 | 0.8 | 0.8 | Compare two fixed-point numbers for "less than or equal to". If ACCUM 2 ACCUM 1, the RLO is "1". CC 1/CC 0 are affected. |
| **Block Call Operations** | | | | | | | | | | | | | |
| JU | OB | | | | | N | N | Y | 6.7 | 6.7 | 3.4 | 3.4 | Jump unconditionally to an organization block. |
| JU | PB | | | | | N | N | Y | 6.7 | 6.7 | 3.4 | 3.4 | Jump unconditionally to a program block. |
| JU | FB | | | | | N | N | Y | 6.7 | 6.7 | 3.4 | 3.4 | Jump unconditionally to a function block. |
| JU | SB | | | | | N | N | Y | 6.7 | 6.7 | 3.4 | 3.4 | Jump unconditionally to a sequence block. |
| JC | OB | | | | | Y | Y [1] | Y | 6.7<br>1.7 | 6.7<br>1.7 | 3.4<br>0.9 | 3.4<br>0.9 | Jump conditionally to an organization block.<br>Time applies for RLO=1/RLO=0 |
| JC | PB | | | | | Y | Y [1] | Y | 6.7<br>1.7 | 6.7<br>1.7 | 3.4<br>0.9 | 3.4<br>0.9 | Jump conditionally to a program block.<br>Time applies for RLO=1/RLO=0 |
| JC | FB | | | | | Y | Y [1] | Y | 6.7<br>1.7 | 6.7<br>1.7 | 3.4<br>0.9 | 3.4<br>0.9 | Jump conditionally to a function block.<br>Time applies for RLO=1/RLO=0 |
| JC | SB | | | | | Y | Y [1] | Y | 6.7<br>1.7 | 6.7<br>1.7 | 3.4<br>0.9 | 3.4<br>0.9 | Jump conditionally to a sequence block.<br>Time applies for RLO=1/RLO=0 |

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? | | | Typical Execution time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Block Call Operations (Continued)** | | | | | | | | | | | | | |
| Q | DB | | | | | N | N | N | 3.6 | 3.6 | 1.8 | 1.8 | Call a data block. |
| I | DB | | | | | N | N | N | 270 | 270 | 270 | 270 | Generate a data block. The number of data words in the block must be stored in ACCUM 1. |
| **Return Operations** | | | | | | | | | | | | | |
| BE | | | | | | N | N | Y | 5 | 5 | 2.5 | 2.5 | Block end (termination of a block) |
| BEC | | | | | | Y | Y[1] | Y | 5<br>1.7 | 5<br>1.7 | 2.5<br>0.9 | 2.5<br>0.9 | Block end, conditional<br>Time applies for RLO=1/RLO=0 |
| BEU | | | | | | N | N | Y | 5 | 5 | 2.5 | 2.5 | Block end, unconditional |
| **"No" Operations** | | | | | | | | | | | | | |
| NOP 0 | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | No operation (all bits reset) |
| NOP 1 | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | No operation (all bits set) |
| **Stop Operation** | | | | | | | | | | | | | |
| STP | | | | | | N | N | N | 50 | 50 | 50 | 50 | Stop: scanning cycle is still completed. Error ID "STS" is set in the ISTACK. |
| **Display Generation Operations** | | | | | | | | | | | | | |
| BLD 130 | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Display generation operation for the programmer: carriage return generates blank line. |
| BLD 131 | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Display generation operation for the programmer: switch over to statement list (STL) |
| BLD 132 | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Display generation operation for the programmer: switch over to control system flowchart (CSF) |

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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Display Generation Operations (Continued)** | | | | | | | | | | | | | |
| BLD 133 | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Display generation operation for the programmer: switch over to ladder diagram (LAD) |
| BLD 255 | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Display generation operation for the programmer: segment termination |

## 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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Boolean Logic Operations** | | | | | | | | | | | | | |
| A= | Formal operand | | | | | N | Y | N | 160 * | 160 * | 160 * | 3.6 * | AND operation: scan formal operand for "1". (Data type: BI) |
| | • | • | • | • | • | | | | | | | | |
| AN= | Formal operand | | | | | N | Y | N | 163 * | 163 * | 163 * | 3.6 * | AND operation: scan formal operand for "0". (Data type: BI) |
| | • | • | • | • | • | | | | | | | | |
| O= | Formal operand | | | | | N | Y | N | 164 * | 164 * | 164 * | 3.6 * | OR operation: scan formal operand for "1". (Data type: BI) |
| | • | • | • | • | • | | | | | | | | |
| ON= | Formal operand | | | | | N | Y | N | 165 * | 165 * | 165 * | 3.6 * | OR operation: scan formal operand for "0". (Data type: BI) |
| | • | • | • | • | • | | | | | | | | |
| AW | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Combine contents of ACCUM 2 and ACCUM 1 through logic AND (word operation). Result is stored in ACCUM 1. CC 1/CC 0 are affected. |
| OW | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Combine contents of ACCUM 2 and ACCUM 1 through logic OR (word operation). Result is stored in ACCUM 1. CC 1/CC 0 are affected. |
| XOW | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Combine contents of ACCUM 2 and ACCUM 1 through EXCLUSIVE OR (word operation). Result is stored in ACCUM 1. CC 1/CC 0 are affected. |

***** 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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Bit Operations** | | | | | | | | | | | | | |
| TB | | | | • | • | N | Y | N | 143 | 143 | 143 | 143 | Test a timer or counter word bit for "1". |
| TB | D | | | | | N | Y | N | 155 | 155 | 155 | 155 | Test a data word bit for "1". |
| TB | RS | | | | | N | Y | N | 141 | 141 | 141 | 141 | Test a data word bit in the system data range for "1". |
| TBN | | | | • | • | N | Y | N | 143 | 143 | 143 | 143 | Test a timer or counter word bit for "0". |
| TBN | D | | | | | N | Y | N | 159 | 159 | 159 | 159 | Test a data word bit for "0". |
| TBN | RS | | | | | N | Y | N | 139 | 139 | 139 | 139 | Test a data word bit in the system data range for "0". |
| SU | | | | • | • | N | N | Y | 143 | 143 | 143 | 143 | Set a timer or counter word bit unconditionally. |
| SU | D | | | | | N | N | Y | 159 | 159 | 159 | 159 | Set a data word bit unconditionally. |
| RU | | | | • | • | N | N | Y | 143 | 143 | 143 | 143 | Reset a timer or counter word bit unconditionally. |
| RU | D | | | | | N | N | Y | 158 | 158 | 158 | 158 | Reset a data word bit unconditionally. |
| **Set/Reset Operations** | | | | | | | | | | | | | |
| S= | Formal operand | | | | | Y | N | Y | 150* | 150* | 150* | 3.6* | Set a formal operand, (with RLO =1). (Data type: BI) |
| | • | • | • | | | | | | | | | | |
| RB= | Formal operand | | | | | Y | N | Y | 150* | 150* | 150* | 3.6* | Reset a formal operand, (with RLO =1). (Data type: BI) |
| | • | • | • | | | | | | | | | | |
| RD= | Formal operand | | | | | Y | N | Y | 146* | 146* | 146* | 3.6* | Reset a formal operand (digital) (with RLO =1). |
| | | | | • | • | | | | | | | | |
| == | Formal operand | | | | | N | N | Y | 150* | 150* | 150* | 3.6* | The value of the RLO is assigned to the status of the formal operand (Data type: BI) |
| | • | • | • | | | | | | | | | | |

***** 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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Timer and Counter Operations** | | | | | | | | | | | | | |
| FR | | | | • | • | Y | N | Y | 3.7 | 3.7 | 3.7 | 1.9 | Enable a timer/counter for cold restart. If RLO="1", - "FR T" restarts the timer - "FR Z" sets, decrements, or increments the counter |
| FR= | Formal operand | | | • | • | Y | N | Y | 144 * | 144 * | 144 * | 3.6 * | Enable formal operand (timer/counter) for cold restart (for detailed description, see "FR" operation). |
| SP= | Formal operand | | | • | | Y | N | Y | 144 * | 144 * | 144 * | 3.6 * | Start a timer (formal operand) as pulse with the value stored in ACCUM 1. |
| SR= | Formal operand | | | • | | Y | N | Y | 144 * | 144 * | 144 * | 3.6 * | Start an on-delay timer (formal operand) with the value stored in ACCUM 1. |
| SEC= | Formal operand | | | • | • | Y | N | Y | 144 * | 144 * | 144 * | 3.6 * | Start a timer (formal operand) as extended pulse with the value stored in ACCUM 1, or set a counter (formal operand) with the next indicated count value. |
| SSU= | Formal operand | | | • | • | Y | N | Y | 144 * | 144 * | 144 * | 3.6 * | Start a stored on-delay timer (formal operand) with the value stored in ACCUM 1, or increment a counter (formal operand). |
| SFD= | Formal operand | | | • | • | Y<br>Y | N | Y | 144 * | 144 * | 144 * | 3.6 * | Start an off-delay timer ( ) (formal operand) with the value stored in ACCUM 1, or decrement a counter ( ) (formal operand). |
| **Load and Transfer Operations** | | | | | | | | | | | | | |
| L= | Formal operand | | | | | N | N | N | 147 * | 147 * | 147 * | 3.6 * | Load the value of the formal operand into ACCUM 1 (Data type: BY, W; additional actual operands: DL, DR, DW). |
| L | RS | | | | | N | N | N | 89 | 89 | 89 | 89 | Load a word from the system data range into ACCUM 1. |
| LC= | Formal operand | | | • | • | N | N | N | 145 * | 145 * | 145 * | 3.6 * | Load the value of the formal operand in BCD code into ACCUM 1. |
| LW= | Formal operand | | | | | N | N | N | 124 * | 124 * | 124 * | 3.6 * | Load a formal operand bit pattern into ACCUM 1 (parameter type: D; Data type: KC, KF, KH, KM, KS, KT, KY). |
| T= | Formal operand | | | | | N | N | N | 148 * | 148 * | 148 * | 3.6 * | Transfer the contents of ACCUM 1 to the formal operand (Data type: BY, W; additional actual operands: DL, DR, DW). |

***** 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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Conversion Operations** | | | | | | | | | | | | | |
| CFW | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Form the one's complement of ACCUM 1. |
| CSW | | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Form the two's complement of ACCUM 1. CC 1/CC 0 and OV are affected. |
| **Shift Operations** | | | | | | | | | | | | | |
| SLW | Parameter n=0 to 15 | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Shift the contents of ACCUM 1 to the left by the value specified in the parameter. Unassigned positions are padded with zeros. CC 1/CC 0 are affected. |
| SRW | Parameter n=0 to 15 | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Shift the contents of ACCUM 1 to the right by the value specified in the parameter. Unassigned positions are padded with zeros. CC 1/CC 0 are affected. |
| **Jump Operations** | | | | | | | | | | | | | |
| JU= | Symbolic address maximum 4 characters | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Jump unconditionally to the symbolic address. |
| JC= | Symbolic address maximum 4 characters | | | | | Y | Y[1] | Y | 1.6 | 1.6 | 0.8 | 0.8 | Jump conditionally to the symbolic address. (If the RLO is "0", it is set to "1"). |
| JZ= | Symbolic address maximum 4 characters | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Jump if the result is zero. The jump is made only if CC 1=0 and CC 0=0. The RLO is not changed. |
| JN= | Symbolic address maximum 4 characters | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Jump if the result is not zero. The jump is made only if CC 1 CC 0 . The RLO is not changed. |
| JP= | Symbolic address maximum 4 characters | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Jump if the result>0. The jump is made only if CC 1=1 und CC 0=0. The RLO is not changed. |
| JM= | Symbolic address maximum 4 characters | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Jump if the result <0. The jump is made only if CC 1=0 and CC 0=1. The RLO is not changed. |
| JO= | Symbolic address maximum 4 characters | | | | | N | N | N | 1.6 | 1.6 | 0.8 | 0.8 | Jump on overflow. The jump is made only 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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Other Operations** | | | | | | | | | | | | | |
| IA | | | | | | N | N | N | 55 | 55 | 55 | 55 | Disable interrupt. Input/output interrupt and timer OB processing is disabled. |
| RA | | | | | | N | N | N | 55 | 55 | 55 | 55 | Enable interrupt. This operation cancels the effect of IA. |
| D | | | | | | N | N | N | 1.7 | 1.7 | 0.9 | 0.9 | Decrement the low byte (bits 0 to 7) of ACCUM 1 by the value n (n=0 to 255). |
| I | | | | | | N | N | N | 1.7 | 1.7 | 0.9 | 0.9 | Increment the low byte (bits 0 to 7) of ACCUM 1 by the value n (n=0 to 255). |
| DO= | Formal operand | | | | | N | N | N | 170 * | 170 * | 170 * | 3.6 * | Process a block. (Only C DB, JU PB, JU FB, JU SB and JU OB can be substituted). Actual operands: C DB, JU PB, JU FB, JU SB and JU OB. |
| | • | • | • | • | • | | | | | | | | |
| DO | DW ** | | | | | N | N | N | 162 * | 162 * | 162 * | 3.6 * | Process data word. The next operation is combined through logic OR with the parameter specified in the data word and executed**. |
| DO | FW ** | | | | | N | N | N | 134 * | 134 * | 134 * | 2.6 * | Process flag word. The next operation is combined through logic OR 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,SR T, SP T, SS T, SE T; D, I;
FR C, R C, S C, CR 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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Set Operations** | | | | | | | | | | | | | |
| SU | RS | | | | | N | N | Y | 142 | 142 | 142 | 142 | Set bit in system data range unconditionally. |
| RU | RS | | | | | N | N | Y | 142 | 142 | 142 | 142 | Reset bit in system data range unconditionally. |
| **Load and Transfer Oprations** | | | | | | | | | | | | | |
| LIR | 0 ( ACCUM 1)<br>2 ( ACCUM 2) | | | | | N | N | N | 126* | 126* | 126* | 4.5** | Load the contents of a memory word (addressed by ACCUM 1) indirectly into the register (0: ACCUM 1; 2: ACCUM 2). [1] |
| TIR | 0 ( ACCUM 1)<br>2 ( ACCUM 2) | | | | | N | N | N | 105* | 105* | 105* | 4.5** | Transfer the contents of the accumulator indirectly to the memory word (addressed by ACCUM 1) (0: ACCUM 1; 2: ACCUM 2) [1] |
| LDI | A1 ( ACCUM 1)<br>A2 ( ACCUM 2) | | | | | N | N | N | - | - | - | 126 | Load the contents of a memory word (addressed by ACCUM 1) indirectly into ACCUM 1 or ACCUM 2 (A1=ACCUM 1, A2=ACCUM 2). [2] |
| TDI | A1 ( ACCUM 1)<br>A2 ( ACCUM 2) | | | | | N | N | N | - | - | - | 105 | Transfer the contents of the accumulator indirectly into the memory word (addressed by ACCUM 1) (A1=ACCUM 1, A2=ACCUM 2) [2] |
| TNB | Parameter<br>n=0 to 255 | | | | | N | N | N | * 68 + 34 · n | * 68 + 34 · n | * 68 + 34 · n | * 2.9+ n(1.7 +*) | Transfer a block byte by byte (number of bytes 0 to 255).<br>End address source: ACCUM 2<br>End address target: ACCUM 2 |
| T | RS | | | | | N | N | N | 75 | 75 | 75 | 75 | Transfer a word to the system data range. |
| **Jump Operation** | | | | | | | | | | | | | |
| JUR | | | | | | N | N | N | 105 | 105 | 105 | 105 | Jump randomly within a function block (jump displacement: - 32768 to+32767). |
| **Arithmetic Operations** | | | | | | | | | | | | | |
| ADD | BF | | | | | N | N | N | 57 | 57 | 57 | 57 | Add byte constant (fixed point) to ACCUM 1. |
| ADD | KF | | | | | N | N | N | 90 | 90 | 90 | 90 | Add fixed-point constant (word) to ACCUM 1. |

1 In the case of CPU 944 access to memory bank 1
2 In the case of CPU 944 access to memory bank 2
* When accessing the I/0 area, the relevant timeouts for each byte access must be added.
** + 2 x ready delay time of the referenced I/O modules

☐ 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? | | | Typical Execution Time in µsec. | | | | Function |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | I | Q | F | T | C | 1 | 2 | 3 | CPU 941 | CPU 942 | CPU 943 | CPU 944 | |
| **Other Operations** | | | | | | | | | | | | | |
| DI | Formal operand | | | | | N | N | N | 174* | 174* | 174* | 174* | Process via a formal operand (indirectly). The number of the formal operand is in ACCUM 1. |
| | • | • | • | • | • | | | | | | | | |
| STS | | | | | | N | N | N | 50 | 50 | 50 | 50 | Stop operation. Program processing is interrupted immediately after this operation. |
| TAK | | | | | | N | N | N | 80 | 80 | 80 | 80 | Swap the contents of ACCUM 1 and ACCUM 2. |

* plus execution time of the substituted operation

## 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 | FB |
| 0 | B | $0_a$ | $0_a$ | | | | | T | FB |
| 0 | C | $0_d$ | $0_d$ | | | | | LC | T |
| 0 | D | $0_i$ | $0_i$ | | | | | JO= | |
| 0 | E | $0_c$ | $0_c$ | | | | | LC= | |
| 0 | F | $0_c$ | $0_c$ | | | | | O= | |
| 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 | $0_g$ | $0_g$ | | | | | T | DL |
| 2 | 4 | $0_d$ | $0_d$ | | | | | 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 | $0_c$ | $0_c$ | | | | | SSU= | |
| 2 | F | $0_c$ | $0_c$ | | | | | ON= | |
| 3 | 0 | 0 | 1 | $0_e$ | $0_e$ | $0_e$ | $0_e$ | L | KC |
| 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 | KS |
| 3 | 0 | 2 | 0 | $0_e$ | $0_e$ | $0_e$ | $0_e$ | L | KY |
| 3 | 0 | 4 | 0 | $0_e$ | $0_e$ | $0_e$ | $0_e$ | L | KH |
| 3 | 0 | 8 | 0 | $0_e$ | $0_e$ | $0_e$ | $0_e$ | 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 | PB |
| 5 | 8 | 0 | 0 | $0_e$ | $0_e$ | $0_e$ | $0_e$ | ADD | KF |
| 5 | 9 | 0 | 0 | | | | | -F | |
| 5 | C | $0_o$ | $0_o$ | | | | | S | C |
| 5 | D | $0_f$ | $0_f$ | | | | | JC | SB |

| Machine Code | | | | | | | | Operation | Operand |
|---|---|---|---|---|---|---|---|---|---|
| B0 | | B1 | | B2 | | B3 | | | |
| L | R | L | R | L | R | L | R | | |
| 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 | 8 | 0 | B | | | | | LDI | A1 |
| 6 | 8 | 0 | F | | | | | TDI | A1 |
| 6 | 8 | 2 | B | | | | | LDI | A2 |
| 6 | 8 | 2 | F | | | | | TDI | A2 |
| 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 | $0_b$ | $0_o$ | $0_o$ | 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 | $0_b$ | $0_d$ | $0_d$ | 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 | $0_a$ | $0_a$ | | | | | L | PB |
| 7 | 3 | $0_a$ | $0_a$ | | | | | T | PB |
| 7 | 5 | $0_f$ | $0_f$ | | | | | JU | PB |
| 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 | $0_b$ | $0_a$ | $0_a$ | | | | | A | F |
| 8 | $8_b$ | $0_a$ | $0_a$ | | | | | O | F |
| 9 | $0_b$ | $0_a$ | $0_a$ | | | | | S | F |
| 9 | $8_b$ | $0_a$ | $0_a$ | | | | | = | F |
| A | $0_b$ | $0_a$ | $0_a$ | | | | | AN | F |
| A | $8_b$ | $0_a$ | $0_a$ | | | | | ON | F |
| B | $0_b$ | $0_a$ | $0_a$ | | | | | R | F |
| 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 the Indices**

**a + byte address**
**b + bit address**
**c + parameter address**
**d + timer number**
**e + constant**
**f + block number**
**g + word address**
**h + number of shifts**
**i + relative jump address**
**k + register address**
**l + block length in bytes**
**m + jump displacement (16 bit)**
**n + value**
**o + counter number**

## B Maintenance


B.1 Changing Fuses . . . . . B. - 1

B.2 Installing or Changing Battery . . . . . B - 1
B.2.1 Removing the Battery . . . . . B. - 2
B.2.2 Installing the Battery . . . . . B. - 2
B.2.3 Battery Disposal . . . . . B - 3

B.3 Changing the Fan Filter . . . . . B. - 3

B.4 Replacing the Fan Motor . . . . . B. - 4

**Figures**

B-1. Opening the Battery Compartment . . . . . B - 2
B-2. Changing the Fan Filter . . . . . B .- 3

# B Maintenance

Proper functioning of the programmable controller can only be guaranteed if the electronic components have not been interfered with.
This appendix describes the maintenance jobs you can perform on your programmable controller.
These are

- Changing the fuses
- Installing and changing the batteries
- Changing the fan filter
- Replacing the fan motor

## B.1 Changing Fuses

For the output modules with red LED indicators for fuse failure, you can remove the fuses with a screwdriver (maximum width 3 mm). Swing the front connectors out to access the fuses. Fuse specifications are noted on the inside of the front doors.

## B.2 Installing or Changing Battery

The power supply modules 6ES5 951-7LB21/7NB21 are provided with a backup battery. Use a 3.4 V / 5 Ah lithium battery (Order No. 6EW1 000-7AA; size C) for backup in those power supply modules. Its service life for continuous backup is at least two years (one year when using CPs).
The power supply modules 6ES5 951-7LD21/7ND41/7ND51 are provided with two backup batteries. Use a 3.6 V / 1.75 Ah lithium battery each (Order No. 6ES5 980-0AE11; size AA) for backup in those power supply modules. Its service life for continuous backup is at least one year.

**Changing Battery in Power Supply Modules with 2 Backup Batteries**

- Since the second battery takes over the backup function you can change the discharged battery without any problems.
- After changing of the battery, the backup function remains with the second battery. Only when this battery is discharged, the new battery takes over the backup functions.

**Note**

If you install or change a battery when the PLC is shut off and there is no external voltage supply, perform an Overall Reset on the CPU afterwards. Otherwise, the CPU cannot go into the RUN mode.

## B.2.1 Removing the Battery

Proceed as follows:

1. Open the battery compartment as follows (see Figure B.1)
   Press the slide down.
   Swing the battery compartment door out and down.
2. Removing the battery
   Remove the battery by pulling the end of the plastic ribbon out. The battery slides out of its clamp and falls out.
3. Closing the battery compartment door
   Close the battery compartment door by swinging it back into place. Latch it with the slide.

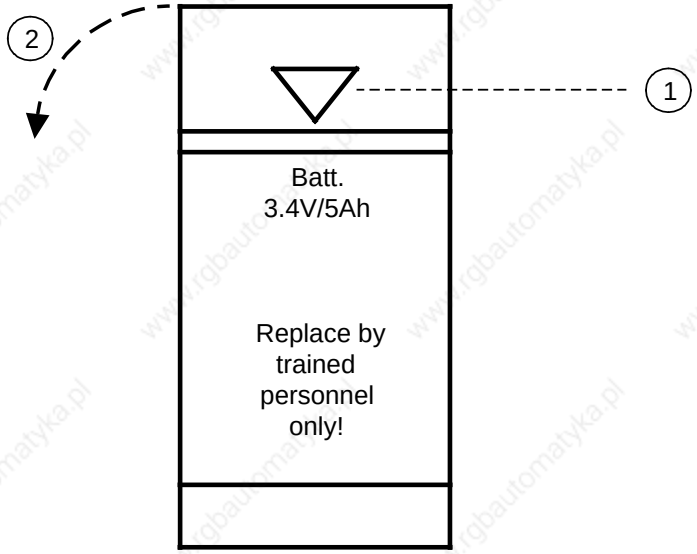



**Figure B-1. Opening the Battery Compartment**

## B.2.2 Installing the Battery

To install a battery, proceed as described below:

1. Open the battery compartment door (see Figure B.1)
   Press the slide down and
   swing the battery compartment door out and down.

Install the battery after noting the following points:

- The poles are indicated on the back of the battery compartment.
- The plastic ribbon should be to the left of the battery so that its end stays in a freely accessible position.
- Before using a lithium battery, you should depassivate it by loading it with 100 ohms for two hours.

2. Install the battery
3. Close the battery compartment door.
   Close the door of the battery compartment by swinging it back into place. Latch it with the slide.

### B.2.3 Battery Disposal

Note the warning below and dispose of used batteries carefully!

**Warning**

Improper handling can cause a lithium battery to catch fire and explode. Do not recharge or disassemble a lithium battery.
Keep it away from water and open flame. Do not expose it to temperatures greater than 100 ° C!

## B.3 Changing the Fan Filter

Under the fan is a filter (Order No. 6ES5 981-0JA11) to keep the electronic components and the printed circuit boards in the modules clean. As preventive maintenance, change this filter regularly according to the degree of air pollution in the PLC's environment.

To change the filter, proceed as described below (see Figure B.2):
1. Pull out the dirty filter using the two handles .
2. Place the new filter in the guide tracks and push it back.

**Note**

You can change the filter while the PLC is operating.

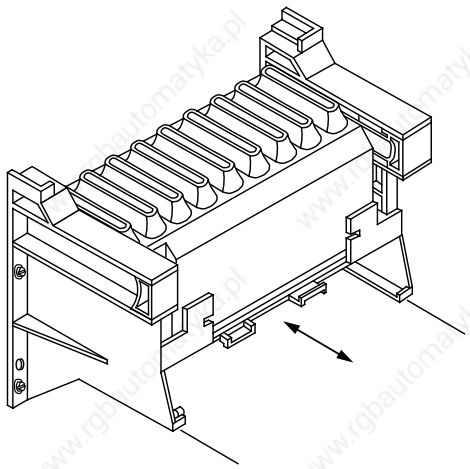


**Figure B-2. Changing the Fan Filter**

## B.4 Replacing the Fan Motor

The fan motors can be exchanged in all fan subassemblies of the S5-115U programmable controller.
For this purpose, Siemens offers a fan replacement package (Order No. 6ES5 988-7NA11).

This package contains the following:
- Fan motor
- Plug-in connector
- Repair instructions

Since the repair instructions form part of the fan replacement package, the removal and installation of the fan motor is not described in this section.

## C Module Slots


C.1 Connector Pin Assignment for Power Supply Module . . . . . C - 1

C.2 Connector Pin Assignment of the CPUs . . . . . C - 2

C.3 Connector Pin Assignment for CPs and Intelligent I/Os . . . . . C - 3

C.4 Connector Pin Assignment for Digital and Analog Input/Output Modules . . . . . C - 4

C.5 Connector Pin Assignment for Interface Modules . . . . . C - 5

C.5.1 Connector Pin Assignment of the Symmetrical and Serial EU Interface Modules . . . . . C - 5

C.5.2 Connector Pin Assignment of the Symmetrical and Serial CC Interface Modules . . . . . C - 6

C.5.3 Connector Pin Assignment of the IM 305/IM 306 Interface Modules . . . . . C - 7

C.6 Connector Pin Assignment of the ER 701-3 Mounting Rack . . . . . C - 8

C.7 Legend for Connector Pin Assignment . . . . . C - 11

# C Module Slots

## C.1 Connector Pin Assignment for Power Supply Module

Upper connector

| | a | b |
|---|---|---|
| 1 | M | |
| 2 | +5 V | |
| 3 | +5 V | |
| 4 | +5 V | |
| 5 | +5 V | |
| 6 | +5 V | |
| 7 | +5 V | |
| 8 | +5.2 V | |
| 9 | M | |
| 10 | M | |
| 11 | UBATT | |
| 12 | M | |
| 13 | $\overline{\text{HOLD}}$* | |
| 14 | M | |
| 15 | $\overline{\text{RESET}}$A | |
| 16 | M | |
| 17 | RESET | |
| 18 | M | |
| 19 | $\overline{\text{BAU}}$ | |
| 20 | M | |
| 21 | | |
| 22 | $\overline{\text{HOLDA1}}$* | |
| 23 | $\overline{\text{NAU}}$ | |
| 24 | $\overline{\text{HOLDA2}}$* | |
| 25 | $\overline{\text{PEU}}$ | |
| 26 | $\overline{\text{HOLDA3}}$* | |
| 27 | $\overline{\text{DSI}}$** | |
| 28 | M | |
| 29 | +24 V | |
| 30 | M24 V | |
| 31 | | |
| 32 | M | |

Lower connector (only on CC 2/3 and EU 2/3 )

| | a | b |
|---|---|---|
| 1 | M | |
| 2 | +5 V | |
| 3 | +5 V | |
| 4 | +5 V | |
| 5 | +5 V | |
| 6 | M | |
| 7 | M | |
| 8 | M | |
| 9 | M | |
| 10 | $\overline{\text{NAU}}$ | |
| 11 | M | |
| 12 | $\overline{\text{BAU}}$ | |
| 13 | M | |
| 14 | $\overline{\text{RESET}}$A | |
| 15 | M | |
| 16 | $\overline{\text{PEU}}$ | |
| 17 | M | |
| 18 | $\overline{\text{HOLDA3}}$* | |
| 19 | $\overline{\text{HOLDA2}}$* | |
| 20 | $\overline{\text{HOLDA1}}$* | |
| 21 | $\overline{\text{HOLD}}$* | |
| 22 | | |
| 23 | | |
| 24 | | |
| 25 | | |
| 26 | | |
| 27 | | |
| 28 | | |
| 29 | +24 V | |
| 30 | M24 V | |
| 31 | | |
| 32 | M | |

* Only on 7/15 A power supply module
** Not on CR 700-1, only with 7/15 A power supply

## C.2 Connector Pin Assignment of the CPUs

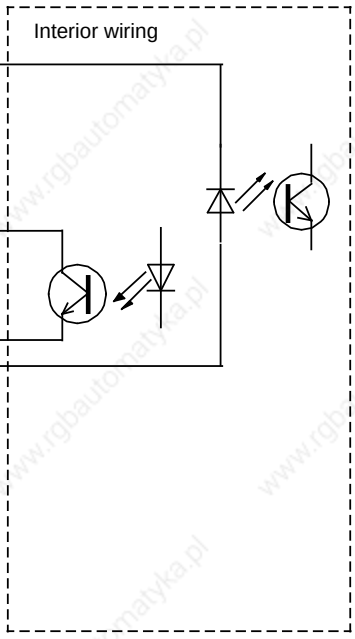
CPU slot
Upper connector

| | z | b | d | f |
|---|---|---|---|---|
| 2 | +5 V | M | +5.2 V | +5 V |
| 4 | TAKT | PESP | UBATT | $\overline{F0}$ |
| 6 | RESET | ADB0 | ADB12 | $\overline{F1}$ |
| 8 | $\overline{MRD}$ | ADB1 | ADB13 | $\overline{F2}$ |
| 10 | $\overline{MWR}$ | ADB2 | ADB14 | $\overline{F3}$ |
| 12 | $\overline{RDY}$ | ADB3 | ADB15 | $\overline{F4}$ |
| 14 | DB0 | ADB4 | $\overline{IRA}$ | $\overline{F5}$ |
| 16 | DB1 | ADB5 | $\overline{IRB}$ | $\overline{F6}$ |
| 18 | DB2 | ADB6 | $\overline{IRC}$* | $\overline{ASF}$ |
| 20 | DB3 | ADB7 | $\overline{IRD}$* | $\overline{HOLD}$ |
| 22 | DB4 | ADB8 | $\overline{BAU}$ | $\overline{HOLDA1}$ |
| 24 | DB5 | ADB9 | $\overline{NAU}$ | $\overline{HOLDA2}$* |
| 26 | DB6 | ADB10 | $\overline{PEU}$ | $\overline{HOLDA3}$* |
| 28 | DB7 | ADB11 | | $\overline{PRAL}$ |
| 30 | M24 V | BASP | | +24 V |
| 32 | | M | $\overline{BASPA}$ | $\overline{ASG}$ |

* Only on 7/15A power supply module

Interface assignment of the serial interface

| | |
|---|---|
| 1 | $M_{EXT}$ |
| 2 | TTY IN- |
| 3 | +5.2 V |
| 4 | + 24 V |
| 5 | M |
| 6 | TTY OUT+ |
| 7 | TTY OUT- |
| 8 | $M_{EXT}$ |
| 9 | TTY IN+ |
| 10 | M 24 V |
| 11 | 20 mA |
| 12 | M |
| 13 | 20 mA |
| 14 | + 5.2 V |
| 15 | M |



## C.3 Connector Pin Assignment for CPs and Intelligent I/Os

Slots 0 to 5 (left)*
Upper connector

| | z | b | d |
|---|---|---|---|
| 2 | +5 V | M | +5.2 V |
| 4 | TAKT | PESP | UBATT |
| 6 | RESET | ADB0 | ADB12 |
| 8 | $\overline{\text{MRD}}$ | ADB1 | ADB13 |
| 10 | $\overline{\text{MWR}}$ | ADB2 | ADB14 |
| 12 | $\overline{\text{RDY}}$ | ADB3 | ADB15 |
| 14 | DB0 | ADB4 | $\overline{\text{IRA}}$ |
| 16 | DB1 | ADB5 | $\overline{\text{IRB}}$ |
| 18 | DB2 | ADB6 | $\overline{\text{IRC}}$ |
| 20 | DB3 | ADB7 | $\overline{\text{IRD}}$ |
| 22 | DB4 | ADB8 | $\overline{\text{BAU}}$ |
| 24 | DB5 | ADB9 | $\overline{\text{NAU}}$ |
| 26 | DB6 | ADB10 | $\overline{\text{PEU}}$ |
| 28 | DB7 | ADB11 | $\overline{\text{DSI}}$ *** |
| 30 | M24 V | BASP | +24 V |
| 32 | | M | $\overline{\text{BASPA}}$ |

Lower connector;
only on CC 2 and EU 3

| | z | b | d |
|---|---|---|---|
| 2 | +5 V | M | |
| 4 | | | |
| 6 | | | |
| 8 | | | |
| 10 | | | |
| 12 | | | |
| 14 | $\overline{\text{NAU}}$ | | |
| 16 | $\overline{\text{BAU}}$ | | |
| 18 | $\overline{\text{HOLDAX}}$** | | |
| 20 | $\overline{\text{HOLD}}$ | | |
| 22 | | | |
| 24 | | | |
| 26 | | | |
| 28 | | | |
| 30 | M+24 V | M+24 V | |
| 32 | +24 V | M | |

* On slot 0 only in CC 0 and CC 1
On slot 0 to 5 only in CC 2
On slots 0a to 5a only in CC 3
0b to 2b
On slots 0a to 6a only in EU 3

** X= $\overline{\text{HOLDA1}}$ Slot 0
$\overline{\text{HOLDA2}}$ Slot 1
$\overline{\text{HOLDA3}}$ Slot 2
} only used in the case of the CPU 944
not in EU 3

*** not on the CR 700-1

## C.4 Connector Pin Assignment for Digital and Analog Input/Output Modules

Slots 0 to 8 (right)*****

| | a | b |
|---|---|---|
| 1 | M | |
| 2 | +5V | |
| 3 | PESP | |
| 4 | ADB0 | |
| 5 | RESET | |
| 6 | ADB1 | |
| 7 | $\overline{\text{MRD}}$ | |
| 8 | ADB2 | |
| 9 | $\overline{\text{MWR}}$ | |
| 10 | ADB3 | |
| 11 | $\overline{\text{RDY}}$ | |
| 12 | ADB4 | |
| 13 | DB0 | |
| 14 | ADB5 | |
| 15 | DB1 | |
| 16 | ADB6 | |
| 17 | DB2 | |
| 18 | ADB7 | |
| 19 | DB3 | |
| 20 | ADB8 | |
| 21 | DB4 | |
| 22 | ADB9 | |
| 23 | DB5 | |
| 24 | ADB10 | |
| 25 | DB6 | |
| 26 | ADB11 | |
| 27 | DB7 | |
| 28 | BASP | |
| 29 | $\overline{\text{PRAL}}$ | |
| 30 | M | |
| 31 | $\overline{\text{ASG}}$ | |
| 32 | $\overline{\text{FX}}$****** | |

* Slots 0 to 3 in CC0 — Slots 0 to 5 in EU 0
Slots 0 to 6 in CC1 — Slots 0 to 8 in EU 1
Slots 0a to 6a in CC2 — Slots 0b to 7b in EU 2
Slots 3a to 5a in CC3 — Slots 0b to 7b in EU 3
Slots 0 to 8 in EU1
Slots 0b to 7b in EU2
Slots 0b to 7b in EU3

****** Enabling lines of the individual slots (X=0 to 8)

## C.5 Connector Pin Assignment for Interface Modules

### C.5.1 Connector Pin Assignment of the Symmetrical and Serial EU Interface Modules

**Slot 6 (left) in CC2**
**Slots 6a and 6b in CC3**

Upper connector

| | z | b | d |
|---|---|---|---|
| 2 | +5 V | M | |
| 4 | TAKT | PESP | +5 V |
| 6 | RESET | ADB0 | ADB12 |
| 8 | $\overline{\text{MRD}}$ | ADB1 | ADB13 |
| 10 | $\overline{\text{MWR}}$ | ADB2 | ADB14 |
| 12 | $\overline{\text{RDY}}$ | ADB3 | ADB15 |
| 14 | DB0 | ADB4 | +5V |
| 16 | DB1 | ADB5 | +5V |
| 18 | DB2 | ADB6 | M |
| 20 | DB3 | ADB7 | M |
| 22 | DB4 | ADB8 | M |
| 24 | DB5 | ADB9 | M |
| 26 | DB6 | ADB10 | M |
| 28 | DB7 | ADB11 | M |
| 30 | | BASP | M |
| 32 | M | M | $\overline{\text{BASPA}}$ |

Lower connector

| | z | b | d |
|---|---|---|---|
| 2 | +5 V | M | |
| 4 | | | |
| 6 | | | |
| 8 | | | |
| 10 | | | |
| 12 | +5 V | +5 V | |
| 14 | +5 V | +5 V | |
| 16 | +5 V | +5 V | |
| 18 | $\overline{\text{RESETA}}$ | $\overline{\text{PEU}}$ | |
| 20 | | | |
| 22 | M | M | |
| 24 | M | M | |
| 26 | M | M | |
| 28 | M | M | |
| 30 | M | M | |
| 32 | M | M | |

## C.5.2 Connector Pin Assignment of the Symmetrical and Serial CC Interface Modules

**Slot 7 (left) in EU2/3**

Upper connector

| | z | b | d |
|---|---|---|---|
| 2 | +5 V | M | |
| 4 | | PESP | +5 V |
| 6 | RESET | ADB0 | |
| 8 | $\overline{\text{MRD}}$ | ADB1 | |
| 10 | $\overline{\text{MWR}}$ | ADB2 | |
| 12 | $\overline{\text{RDY}}$ | ADB3 | |
| 14 | DB0 | ADB4 | +5 V |
| 16 | DB1 | ADB5 | +5 V |
| 18 | DB2 | ADB6 | M |
| 20 | DB3 | ADB7 | M |
| 22 | DB4 | ADB8 | M |
| 24 | DB5 | ADB9 | M |
| 26 | DB6 | ADB10 | M |
| 28 | DB7 | ADB11 | M |
| 30 | | BASP | M |
| 32 | | M | $\overline{\text{BASPA}}$ |

Lower connector

| | z | b | d |
|---|---|---|---|
| 2 | +5 V | M | M |
| 4 | | | |
| 6 | | | |
| 8 | | | |
| 10 | | | |
| 12 | +5 V | +5 V | |
| 14 | +5 V | +5 V | |
| 16 | +5 V | +5 V | |
| 18 | $\overline{\text{RESETA}}$ | $\overline{\text{NAU}}$ | |
| 20 | | | |
| 22 | M | M | |
| 24 | M | M | |
| 26 | M | M | |
| 28 | M | M | |
| 30 | M | M | |
| 32 | M | M | |

## C.5.3 Connector Pin Assignment of the IM 305/IM 306 Interface Modules

Upper connector

| | z | b | d |
|---|---|---|---|
| 2 | +5 V | M | +5 V |
| 4 | | PESP | +5 V |
| 6 | RESET | ADB0 | $\overline{\text{RESETA}}$ |
| 8 | $\overline{\text{MRD}}$ | ADB1 | $\overline{\text{F0}}$ |
| 10 | $\overline{\text{MWR}}$ | ADB2 | $\overline{\text{F1}}$ |
| 12 | $\overline{\text{RDY}}$ | ADB3 | $\overline{\text{F2}}$ |
| 14 | DB0 | ADB4 | $\overline{\text{F3}}$ |
| 16 | DB1 | ADB5 | $\overline{\text{F4}}$ |
| 18 | DB2 | ADB6 | $\overline{\text{F5}}$ |
| 20 | DB3 | ADB7 | $\overline{\text{F6}}$ |
| 22 | DB4 | ADB8 | $\overline{\text{F7}}$* |
| 24 | DB5 | ADB9 | $\overline{\text{F8}}$** |
| 26 | DB6 | ADB10 | |
| 28 | DB7 | ADB11 | $\overline{\text{PEU}}$ |
| 30 | M | BASP | $\overline{\text{ASF}}$ |
| 32 | M | M | $\overline{\text{ASG}}$ |

* Only in EU1, EU2 and EU3
** Only in EU1

## C.6 Connector Pin Assignment of the ER 701-3 Mounting Rack

**Power supply**

Upper connector

| | a | b |
|---|---|---|
| 1 | M | |
| 2 | +5 V | |
| 3 | +5 V | |
| 4 | +5 V | |
| 5 | +5 V | |
| 6 | +5 V | |
| 7 | +5 V | |
| 8 | +5.2 V | |
| 9 | M | |
| 10 | M | |
| 11 | UBATT | |
| 12 | M | |
| 13 | | |
| 14 | M | |
| 15 | $\overline{\text{RESETA}}$ | |
| 16 | M | |
| 17 | RESET | |
| 18 | M | |
| 19 | $\overline{\text{BAU}}$ | |
| 20 | M | |
| 21 | | |
| 22 | | |
| 23 | $\overline{\text{NAU}}$ | |
| 24 | | |
| 25 | $\overline{\text{PEU}}$ | |
| 26 | M | |
| 27 | $\overline{\text{DSI}}$ | |
| 28 | | |
| 29 | | |
| 30 | | |
| 31 | | |
| 32 | M | |

Lower connector

| | a | b |
|---|---|---|
| 1 | M | |
| 2 | +5 V | |
| 3 | +5 V | |
| 4 | +5 V | |
| 5 | +5 V | |
| 6 | M | |
| 7 | M | |
| 8 | M | |
| 9 | M | |
| 10 | NAU | |
| 11 | M | |
| 12 | BAU | |
| 13 | M | |
| 14 | $\overline{\text{RESETA}}$ | |
| 15 | M | |
| 16 | | |
| 17 | M | |
| 18 | | |
| 19 | | |
| 20 | | |
| 21 | | |
| 22 | | |
| 23 | | |
| 24 | | |
| 25 | | |
| 26 | | |
| 27 | | |
| 28 | | |
| 29 | +24 V | |
| 30 | M24V | |
| 31 | | |
| 32 | M | |

**Slots 0a to 6a**

Upper connector

| | z | b | d |
|---|---|---|---|
| 2 | +5 V | M | +5.2 V |
| 4 | TAKT | PESP | UBATT |
| 6 | RESET | ADB0 | ADB12 |
| 8 | $\overline{\text{MRD}}$ | ADB1 | ADB13 |
| 10 | $\overline{\text{MWR}}$ | ADB2 | ADB14 |
| 12 | $\overline{\text{RDY}}$ | ADB3 | ADB15 |
| 14 | DB0 | ADB4 | $\overline{\text{IRA}}$ |
| 16 | DB1 | ADB5 | $\overline{\text{IRB}}$ |
| 18 | DB2 | ADB6 | $\overline{\text{IRC}}$ |
| 20 | DB3 | ADB7 | $\overline{\text{IRD}}$ |
| 22 | DB4 | ADB8 | $\overline{\text{BAU}}$ |
| 24 | DB5 | ADB9 | $\overline{\text{NAU}}$ |
| 26 | DB6 | ADB10 | $\overline{\text{PEU}}$ |
| 28 | DB7 | ADB11 | $\overline{\text{DSI}}$ |
| 30 | M24 V | BASP | +24 V |
| 32 | | M | $\overline{\text{BASPA}}$ |

Lower connector

| | z | b | d |
|---|---|---|---|
| 2 | +5 V | M | |
| 4 | | | |
| 6 | | | |
| 8 | | | |
| 10 | | | |
| 12 | | | |
| 14 | $\overline{\text{NAU}}$ | | |
| 16 | $\overline{\text{BAU}}$ | | |
| 18 | | | |
| 20 | | | |
| 22 | | | |
| 24 | | | |
| 26 | | | |
| 28 | | | |
| 30 | M+24 V | M+24 V | |
| 32 | +24 V | M | |

**Slot 7a**

Upper connector

| | z | b | d |
|---|---|---|---|
| 2 | +5 V | M | |
| 4 | | PESP | +5 V |
| 6 | RESET | ADB0 | ADB12 |
| 8 | $\overline{\text{MRD}}$ | ADB1 | ADB13 |
| 10 | $\overline{\text{MWR}}$ | ADB2 | ADB14 |
| 12 | $\overline{\text{RDY}}$ | ADB3 | ADB15 |
| 14 | DB0 | ADB4 | +5 V |
| 16 | DB1 | ADB5 | +5 V |
| 18 | DB2 | ADB6 | M |
| 20 | DB3 | ADB7 | M |
| 22 | DB4 | ADB8 | M |
| 24 | DB5 | ADB9 | M |
| 26 | DB6 | ADB10 | M |
| 28 | DB7 | ADB11 | M |
| 30 | | BASP | M |
| 32 | | M | $\overline{\text{BASPA}}$ |

Lower connector

| | z | b | d |
|---|---|---|---|
| 2 | +5 V | M | |
| 4 | | | M |
| 6 | | | |
| 8 | | | |
| 10 | | | |
| 12 | +5 V | +5 V | |
| 14 | +5 V | +5 V | |
| 16 | +5 V | +5 V | |
| 18 | $\overline{\text{RESETA}}$ | $\overline{\text{NAU}}$ | |
| 20 | | | |
| 22 | M | M | |
| 24 | M | M | |
| 26 | M | M | |
| 28 | M | M | |
| 30 | M | M | |
| 32 | M | M | |

**Slots 0b to 7b**

Upper connector

| | a | b |
|---|---|---|
| 1 | M | |
| 2 | +5 V | |
| 3 | PESP | |
| 4 | ADB0 | |
| 5 | RESET | |
| 6 | ADB1 | |
| 7 | $\overline{\text{MRD}}$ | |
| 8 | ADB2 | |
| 9 | $\overline{\text{MWR}}$ | |
| 10 | ADB3 | |
| 11 | $\overline{\text{RDY}}$ | |
| 12 | ADB4 | |
| 13 | DB0 | |
| 14 | ADB5 | |
| 15 | DB1 | |
| 16 | ADB6 | |
| 17 | DB2 | |
| 18 | ADB7 | |
| 19 | DB3 | |
| 20 | ADB8 | |
| 21 | DB4 | |
| 22 | ADB9 | |
| 23 | DB5 | |
| 24 | ADB10 | |
| 25 | DB6 | |
| 26 | ADB11 | |
| 27 | DB7 | |
| 28 | BASP | |
| 29 | | |
| 30 | M | |
| 31 | $\overline{\text{ASG}}$ | |
| 32 | $\overline{F_0}$ ... $\overline{F_7}$ | |

## C.7 Legend for Connector Pin Assignment

| | |
|---|---|
| +5 V | Supply voltage for all modules |
| M | Ground for+5 V and+5.2 V |
| +5.2 V | Supply voltage for PG 605U and PG 615 |
| +24 V | Supply voltage for 20 mA interface and programming voltage for PG 615 |
| M24 V | Ground for+24 V |
| $U_{BATT}$ | 3.4 V battery voltage for RAM back-up |
| $\overline{\text{RESET}}$ | Reset pulse for all modules |
| $\overline{\text{RESETA}}$ | Reset pulse request (initiates or extends a Reset pulse). |
| $\overline{\text{BAU}}$ | Battery failure; signal is generated if there is no battery plugged in or if the battery is low. |
| $\overline{\text{NAU}}$ | Powerfail; signal is generated shortly before the supply voltage fails. |
| $\overline{\text{PEU}}$ | I/O not ready; signal is generated when power supply in the expansion unit fails. |
| DSI | Data back-up; signal is generted with a delay after $\overline{\text{NAU}}$ and switches the battery-backed RAM to 'Standby' in the case of some modules (hardware-implemented). |
| BASP | Command output disable; signal is generated when the CPU stops. The signal disables the digital output modules. |
| $\overline{\text{MRD}}$ | Memory Read; is generated on every Read access. |
| $\overline{\text{MWR}}$ | Memory Write; is generated on every Write access. |
| $\overline{\text{RDY}}$ | Ready; signal acknowledging $\overline{\text{MRD}}$ or $\overline{\text{MWR}}$ access. |
| PESP | Memory I/O Select; is generated every time peripheral memory is accessed. |
| $\overline{\text{ASF}}$ | Interface module free; the central controller is operated without interface module. A terminating resistor connector must be plugged in instead of the interface module. |
| $\overline{\text{ASG}}$ | Interface module plugged in |
| $\overline{\text{IRA}}$, $\overline{\text{IRB}}$ | Interrupt A, B; hardware interrupt signals from intelligent I/Os. |
| $\overline{\text{PRAL}}$ | Process interrupt; hardware interrupt signal from a digital I/O module. |
| HOLDA1, 2, 3 | S5 bus enable for IP 252 intelligent process control module. (only in the case of direct I/O access) |
| $\overline{\text{HOLD}}$ | S5 bus reserved for the IP 252 intelligent process controller module. (only in the case of direct I/O access) |
| F0 ... 8 | Enable line for I/Os |
| ADB0 ... 15 | Address bus |
| DB0 ... 7 | Data bus |

# D Active and Passive Faults in Automation Equipment / Guidelines for Handling Electrostatic Sensitive Devices

# D Active and Passive Faults in Automation Equipment / Guidelines for Handling Electrostatic Sensitive Devices

## Active and Passive Faults in Automation Equipment

- Depending on the particular task for which the electronic automation equipment is used, both **active** as well as **passive** faults can result in a **dangerous** situation. For example, in drive control, an active fault is generally dangerous because it can result in an unauthorized startup of the drive. On the other hand, a passive fault in a signalling function can result in a dangerous operating state not being reported to the operator.

- The differentiation of the possible faults and their classification into dangerous and non-dangerous faults, depending on the particular task, is important for all safety considerations in respect to the product supplied.

**Warning**

In all cases where a fault in automation equipment can result in severe personal injury or substantial property damage, i.e., where a dangerous fault can occur, additional external measures must be taken or equipment provided to ensure or force safe operating conditions even in the event of a fault (e.g., by means of independent limit monitors, mechanical interlocks, etc.).

**Procedures for Maintenance and Repair**

If you are carrying out measurement or testing work **on a PLC**, you must adhere to the rules and regulations contained in the "VGB 4.0 Accident Prevention Regulations" of the German employers liability assurance association ("Berufsgenossenschaften"). Pay particular attention to § 8, "Permissible exceptions when working on live parts".

Do not attempt to repair an item of automation equipment yourself. Such repairs may only be carried out by **Siemens service personnel or repair shops Siemens has authorized to carry out such repairs.**

The information in this manual is checked regularly for updating and correctness and may be modified without prior notice. The information contained in this manual is protected by copyright. Photocopying and translation into other languages is not permitted without express permission from Siemens.

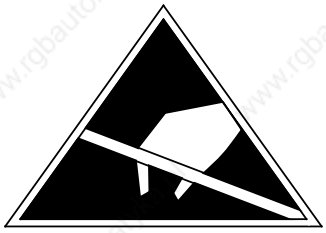
# Guidelines for Handling Electrostatic Sensitive Devices (ESD)

**What is ESD?**

All electronic modules are equipped with large-scale integrated ICs or components. Due to their design, these electronic elements are very sensitive to overvoltages and thus to any electrostatic discharge.

These **E**lectrostatic **S**ensitive **D**evices are commonly referred to by the abbreviation **ESD**.

Electrostatic sensitive devices are labelled with the following symbol:



**Caution**

Electrostatic sensitive devices are subject to voltages that are far below the voltage values that can still be perceived by human beings. These voltages are present if you touch a component module without previously being electrostatically discharged. In most cases, the damage caused by an overvoltage is not immediately noticeable and results in damage only after a prolonged period of operation.

**Electrostatic charging of objects and persons**

Every object with no conductive connection to the electrical potential of its surroundings can be charged electrostatically. In this way, voltages up to 15000 V can build up whereas minor charges, i.e. up to 100 V, are not relevant.

**Examples:**

| | |
|---|---|
| • Plastic covers | up to 5000 V |
| • Plastic cups | up to 5000 V |
| • Plastic-bound books and notebooks | up to 8000 V |
| • Desoldering device with plastic parts | up to 8000 V |
| • Walking on plastic flooring | up to 12000 V |
| • Sitting on a padded chair | up to 15000 V |
| • Walking on a carpet (synthetic) | up to 15000 V |

**Limits for perceiving electrostatic discharges**

An electrostatic discharge is

- perceptible from 3500 V
- audible from 4500 V
- visible from 5000 V

A fraction of those voltages is capable of destroying or damaging electronic devices.

Carefully note and apply the protective measures described below to protect and prolong the life of your modules and components.

**General protective measures against electrostatic discharge damage**

- Keep plastics away from sensitive devices. Most plastic materials have a tendency to build up electrostatic charges easily.

- Make sure that the personnel, working surfaces and packaging are sufficiently grounded when handling electrostatic sensitive devices.

- If possible, avoid any contact with electrostatic sensitive devices. Hold modules without touching the pins of components or printed conductors. In this way, the discharged energy cannot affect the sensitive devices.

**Additional precautions for modules without housings**

Note the following measures that have to be taken for modules that are not protected against accidental contact:

- Touch electrostatical sensitive devices only
  - if you wear a wristband complying with ESD specifications or
  - if you use special ESD footwear or ground straps when walking on an ESD floor.

- Persons working on electronic devices should first discharge their bodies by touching grounded metallic parts (e.g. bare metal parts of switchgear cabinets, water pipes, etc.)

- Protect the modules against contact with chargeable and highly insulating materials, such as plastic foils, insulating table tops or clothes made of plastic fibres.

- Place electrostatic sensitive devices only on conductive surfaces:
  - Tables with ESD surfaces
  - Conductive ESD foam plastic (ESD foam plastic is mostly coloured black)
  - ESD bags

- Avoid direct contact of eletrostatic sensitive devices with visual display units, monitors or TV sets (minimum distance to screen > 10 cm).

The following Figure once again illustrates the precautions for handling electrostatically sensitive devices.

a Conductive flooring material
b Table with conductive, grounded surface
c ESD footwear
d ESD smock
e Gounded ESD wristband
f Grounded connection of switchgear cabinet
g Grounded chair

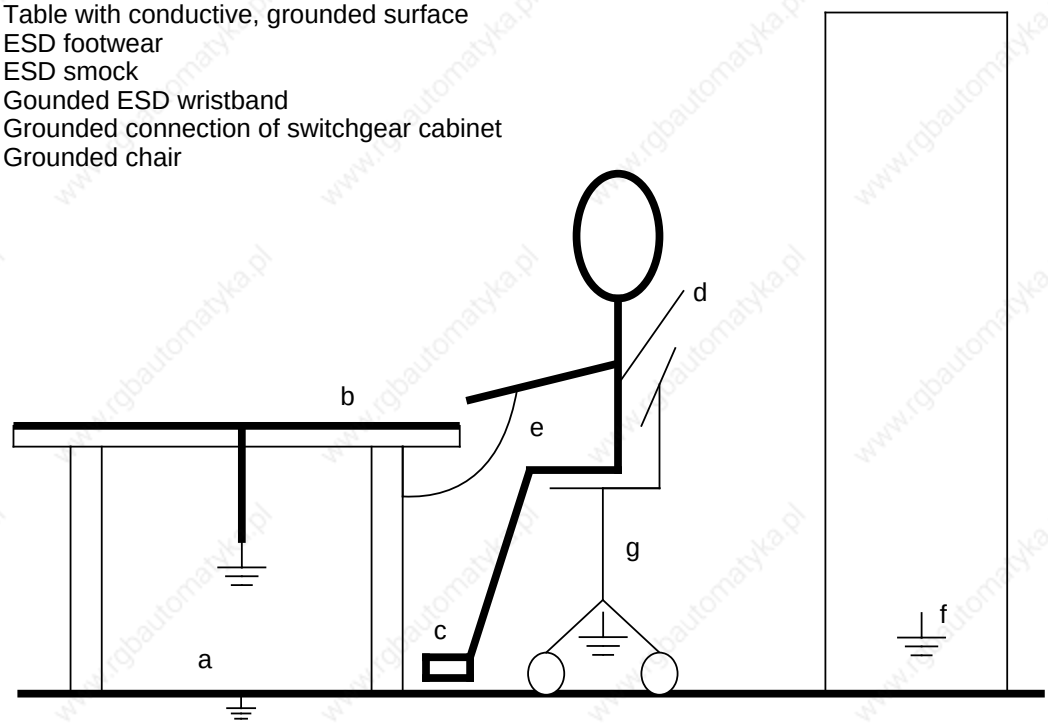



**Figure D-1. ESD Measures**

**Taking measurements and working on ESD modules**

Measurements may be taken on electrostatic sensitive devices only if

- the measuring device is grounded (e.g. via protective conductor) or
- the tip of the isolated measuring tool has previously been discharged (e.g. by briefly touching grounded metal parts).

## E SIEMENS Addresses Worldwide

# E 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 the 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 the Siemens AG
**Berlin**
**Bremen**
**Dortmund**
**Düsseldorf**
**Essen**
**Frankfurt/Main**
**Hamburg**

**Federal Republic of Germany** (continued)
**Hanover**
**Cologne**
**Leipzig**
**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**
**Göteborg**
**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,** Québec
**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.
**México, 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.
**Roswell,** 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**

## List of Abbreviations

# List of Abbreviations

### *List of Abbreviations*

| Abbreviations | Explanation |
|---|---|
| ACCUM 1 | Accumulator 1 (When loading ACCUM 1, the original contents are shifted to ACCUM 2) |
| ACCUM 2 | Accumulator 2 |
| AMPM | DB1 parameter (clock mode- AM/PM representation) |
| ASCII | American Standard Code for Information Interchange |
| BF | Byte constant (fixed-point number) (- 128 to+127) |
| C | Counter (0 to 127) and (0.0 to 127.15) |
| CBR | DB1 parameter (SINEC L1, position of the ”send” coordination byte) |
| CBS | DB1 parameter (SINEC L1, position of the ”receive” coordination byte) |
| CC | Central controller |
| CC0 / CC 1 | Condition code word 0 / condition code word 1 |
| CF | DB1 parameter (correction factor for the integral clock) |
| CLK | DB1 parameter (start of the clock data range) |
| CP | Communications processor |
| CPU | Central processing unit |
| CSF | Control system flowchart in STEP-5 representation |
| D | Date (1 bit) (0.0 to 255.15) |
| DB | Data block (2 to 255) |
| DL | Data word (left byte) (0 to 255) |
| DR | Data word (right byte) (0 to 255) |
| DW | Data word (0 to 255) |
| EMC | Electromagnetic compatibility |
| ERR | DB1 parameter (position of the error code) |
| EU | Expansion unit |
| F | Flag - retentive (0.0 to 127.7)<br>- non-retentive (128.0 to 255.7) |
| FB | Function block (0 to 255) |
| FW | Flag word (0 to 254) |
| FY | Flag byte - retentive (0 to 127)<br>- non-retentive (128 to 255) |

## List of Abbreviations

| Abbreviations | Explanation |
|---|---|
| I | Input<br>(0.0 to 127.7) |
| IB | Input byte<br>(0 to 127) |
| IP | Intelligent I/0 |
| IW | Input word<br>(0 to 126) |
| KB | Constant (1 byte)<br>(0 to 255) |
| KC | Constant (count)<br>(0 to 999) |
| KF | Constant (fixed-point number)<br>(-32768 to +32767) |
| KH | Constant (hexadezimal)<br>(0 to FFFF) |
| KM | Constant (2-byte bit pattern)<br>(random bit pattern: 16 bits) |
| KS | Constant (2 characters)<br>(any 2 alphanumeric characters) |
| KT | Constant (time)<br>(0.0 to 999.3) |
| KY | Constant (2 bytes)<br>(0 to 255 per byte) |
| LAD | Ladder diagram in STEP-5 representation |
| OB | Organization block |
| OHE | DB1 parameter (enable operating hours counter) |
| OHS | DB1 parameter (stop operating hours counter) |
| OP | Operator panel |
| OV | Overflow. This flag is set if,for example, the value range is exceeded in arithmetic operations. |
| PB | Program block (for block call and jump operations)<br>(0 to 255) |
| PB or PY (dependent on programmer) | Peripheral byte<br>(0 to 127) and (128 to 254) |
| PG | Programmer |
| PGN | DB1 parameter (SINEC L1, programmer bus number) |
| PII | Process image of the inputs |
| PIQ | Process image of the outputs |
| PLC | Programmable Logic Controller |
| PRIO | DB1 parameter (fix priority of OB 6) |

## List of Abbreviations

| Abbreviations | Explanation |
|---|---|
| PROT | DB1 parameter (activate software protection) |
| PS | Power supply |
| PW | Peripheral word<br>(0 to 126) |
| Q | Output<br>(0.0 to 127.7) |
| QB | Output byte<br>(0 to 127) |
| QW | Output word<br>(0 to 126) |
| RC | DB1 parameter (set retentive feature of the counter) |
| RDLY | DB1 parameter (restart delay) |
| RF | DB1 parameter (set retentive feature of the flags) |
| RLO | Result of logic operation |
| RM | DB1 parameter (SINEC L1, position of the receive mailbox) |
| RS | System data area<br>- for load operations (supplementary operations) and transfer operations (system operations)<br>(0 to 255)<br>- for bit test and set operations (system operations)<br>(0.0 to 255.15) |
| RT | DB1 parameter (set retentive feature of the timers) |
| SAC | STEP address counter |
| SAV | DB1 parameter (save time - after POWER OFF or RUN STOP) |
| SB | Sequence block<br>CPU 942/943/944 (0 to 255) |
| SET | DB1 parameter (set clock) |
| SF | DB1 parameter (SINEC L1, position of the send mailbox) |
| SFB | DB1 parameter (the number p of the integral FB is replaced by the number q) |
| SLN | DB1 parameter (SINEC L1, slave number) |
| STL | Statement list in STEP-5 representation |
| STP | DB1 parameter (update clock in CPU STOP state) |
| STW | DB1 parameter (position of the status word) |
| T | Timers<br>- for supplementary operations "Bit test" and "Set"<br>(0 to 127) and (0.0 to 127.15) |
| TIS | DB1 parameter (stop timer) |
| WD | DB1 parameter (scan time monitoring) |

# Index

# Index


**A**

Access
- PII 6-8
- PIQ 6-9

Accessories 15-65
Accumulator 2-5, 8-18
Actual operand 7-14
Adapter casing 3-15, 15-65
Address 6-1
- calculation 5-14
- counter 5-11
- setting 6-5

Address allocation
- central processing unit 6-11
- counter 6-18
- flags 6-18
- system data area 6-16
- timer 6-18

Address assignment
- I/O area 6-15

Addressing panel 6-3
Alarm clock 13-12
Analog input modules 10-1
Analog module 1-3
Analog output modules 10-54
Analog value
- digital 10-54
- matching blocks 10-60

Analysis bit 5-2
Arithmetic
- block 11-3
- operation 8-31, 8-70

ASCII
- driver 12-20
- parameter set 12-26

Assigning parameters 12-12
Availability 14-4

**B**

Backup 2-6
- battery 2-16, 2-28, 15-69

Basic operation 8-1, A-4
Battery B-1
Battery backup
- hardware clock 13-12

Binary logic operation 8-59
Binary scaler 8-77
Bit test operation 8-42
Block 7-5
- address list 6-18
- header 7-12
- integral 11-1
- modifying 7-25
- parameter 7-12
- type 7-5, 7-7
- call operation 8-32

Boolean logic operation 8-2
BSTACK 5-15

**C**

Cables
- running of 3-42

Central controller 3-1
Central processing unit 1-3, 2-2, 2-7, 15-14
- address allocation 6-11
- front panel 2-12

Centralized configuration 1-5, 3-7, 3-18
Circuit diagram
- conversion 7-3

Climatic environmental conditions 15-3
Clock
- data area 13-6
- prompt 7-22
- pulse generator 8-78

Code converter 11-2
Coding element 3-14
Cold restart
- characteristics 2-19
- routine 2-14

Common-reference measurement 10-29
Communications
- link 12-38
- processor 1-4, 2-2, 15-59
- system 1-5

Comparison operation 8-30
Compensating box 10-7
Compression 7-25
Condition code generation 8-73
Configuration
- centralized 1-5, 3-8, 3-19
- connection possibilities 3-28
- distributed 1-5, 2-24, 3-8, 3-20

Connector pin assignment
- CPU C-2
- power supply module C-1

CONTROL Block 11-25

Control
- bit 5-7
- circuit 3-33
- logic 6-8
- panel 2-13
- system flowchart (CSF) 7-1
Controller DB 11-33
Conversion
- block 11-2
- operation 8-50
Coordination byte 12-10, 12-23, 12-51
Correction
- algorithm 11-36
- rate algorithm 11-36
- value 13-2, 13-5
Counter 2-4
- address allocation 6-18
- operation 8-25
CPU
- connector pin assignment C-2
CSF control system flowchart
Cyclic program execution 7-20
Cyclic sampling 10-53

**D**

Data
- block 7-16
- handling block 11-5
- interchange 12-1
- setting up 4-5
- traffic 12-21, 12-40
DB1
- default setting 11-46
- initializing 11-46, 11-54
- setting parameters 11-48
Decrement 8-52
Delay time 8-79
Differential measurement 10-29
Digital analog value 10-54
Digital logic operation 8-44
Digital module 1-3, 3-30
Digital output value 10-58
Direct access 6-10
Distributed configuration 1-5, 2-24, 3-7, 3-19
Divider 11-4
Dual port RAM 12-7

**E**

Edge evaluation 8-76
Electrical installation 3-33
Electromagnetic compatibility (EMC) 15-4
Electromagnetic interference 3-37
EMC electromagnetic compatibility
Enable operation 8-41
Environmental conditions
- climatic 15-3
- mechanical 15-3
Equipotential bonding 3-44
Error 5-11
- action
- address 5-11
- analysis 5-1
- diagnostics 5-1
- remedy 5-10
Execution time 2-8
Expansion
- capability 1-4
- unit 3-8

**F**

Fan
- installing 3-16
- subassembly 3-33, 15-67
Fault/error analysis 5-1
FETCH Block 11-24
Field transfer 8-68
Flag 2-4
- address allocation 6-18
FORCE 4-11
FORCE VAR 4-11
Front connector 3-31, 15-66
- pin assignments for AI 460 10-4
- pin assignments for AI 465 10-20
- terminal assignment 15-46, 15-52
Function block 7-11
- integral 11-2
Function select switch
- AI 460 10-15
- AI 465 10-26
Functional unit 2-3
Fuse 15-65, B-1

**G**

GRAPH 5 7-1

**H**

Hardware clock
- battery back-up 13-12

**I**

Increment 8-52
Input module 1-3, 2-2, 15-20, 15-46
Input/output module
- intelligent 1-4
Input value
- digital 10-39
Installation
- electrical 3-35
- fan 3-15
- module 3-13
Integral block 11-1
Integral function block 11-2
Integral real-time clock 13-1
Intelligent input/output module (IP) 1-4, 2-3, 15-54
Interface
- module 2-2, 3-19, 15-60
- serial 2-2, 2-4, 4-12
Interference-free operation 3-46
Interprocessor communication flag 12-1
Interrupt 8-53
- stack 5-2
Interrupt block
- programming 9-1
Interrupt generation
- programmable 9-5
Interrupt response time
- calculating 9-3
Interrupt-driven programming execution 7-22
I/O area
- address assignment 6-15
IP Intelligent input/output module
ISTACK 5-2

**J**

Job status word 11-15, 11-17
Jump operation 5-15, 8-57, 8-69
Jumper setting
- for distributed connection 3-23

**L**

Ladder diagram (LAD) 7-1
LAD ladder diagram
Length word 11-19
Linear programming 7-4
Load
- circuit 3-34
- operation 8-10, 8-40
Loading
- register contents 8-67
- time 8-16
Logic operation
- binary 8-59

**M**

Machine code A-17
Maintenance B-1
Malfunction
- cause 5-18
- report 5-2
Mechanical environmental conditions 15-1
Memory submodule 2-4, 2-28, 5-9
Mode number 12-24, 12-41
Monitor 2-29
Monitoring
- device 1-6
- facilities 3-43
- module 15-64
Mounting rack 2-2, 3-1, 15-3
Multiplier 11-3

**N**

Nesting depth 7-6
Noise immunity 15-2
Number representation 7-26

**O**

Off-delay 8-24
On-delay 8-22
Operand area 7-3
Operating hours counter 13-12
Operating mode
- changing 2-20
- STOP 2-14
Operating system submodule 2-3
Operation
- arithmetic 8-31
- supplementary A-10
- type 7-2
Operations list A-1
Operator
- function 2-12
- panel 1-6, 2-29
Organization block 7-8, 11-32
Output module 1-3, 2-2, 15-31, 15-52
- analog 10-54
- digital 10-58
Overall reset 4-1

**P**

Page 11-6
- addressing 11-20
- frame 12-7
- frame addressing 12-7

Parameter
- assignment error 11-11, 11-19
- set 12-46

PID control algorithm 11-33
PII process input image
Pin assignments
- AI 466 10-29
- for resistance thermometer 10-10, 10-22
- front connector to the AI 460 10-4
- front connector to the AI 465 10-20

PIQ process output image
PLC malfunction 7-23
PLC system
- start-up 4-14

Point-to-point connection 12-16
Potential difference 10-5
Power supply 1-2
Power supply module 2-1, 2-6, 3-29, 15-9
- connector pin assignment C-1

Printer 2-29
Priority 7-21
Process image 6-7
Process input image (PII) 2-4
- access 6-8

Process interrupt 9-1, 9-5
Process output image (PIQ) 2-4
- access 6-9

Process signals
- handling 6-7

Processing operation 8-54
Processor 2-5
Program
- block 7-11
- check 4-8
- error 5-11
- modifying 7-25
- part 7-5
- trace 5-15
- transferring 4-3

Program execution
- cyclic 7-20
- RESTART 7-18
- time-controlled 7-20

Program memory 2-4
- compressing 7-25
- internal 2-4

Programmer 1-6, 2-29
Programming 7-1, 7-8
- error 7-23
- interrupt block 9-1
- interrupt generation 9-5
- linear 7-4
- structured 7-5

Programming execution 7-22
- interrupt-driven

Protection against lightning 3-44
Pulse 8-20

**R**

Range card 10-13, 10-24
Ready delay time 2-24
RECEIVE Block 11-22
Receive mailbox 12-10, 12-18, 12-22, 12-40
Register contents
- loading 8-67
- transferring 8-67

Relay 15-69
Reliability 14-1
RESET block 11-26
Resistance thermometer 10-9, 10-22
- wirebreak signal 10-51

Response time 2-26
RESTART 2-16
Restart characteristics 2-14, 2-18, 7-18
Retentive feature 4-5
RUN 2-17

**S**

Safety 14-5, 15-2
- measures 3-43

Sampling 10-53
- cyclic 10-53
- interval 11-33
- selective 10-53, 10-62

Scan monitoring time 2-27
Scan time
- estimating 2-22
- measuring 2-21

Search 4-8
Selective sampling 10-49, 10-58
SEND block 11-20
Send mailbox 12-10, 12-18, 12-22, 12-40
Sequence block 7-11
Serial interface 2-4, 4-12
Set/reset operation 8-7

Shielding 3-42
Shift operation 8-48
SI 2 interface 12-21, 12-40
Signal
- exchange 12-3, 12-6
- state 6-7
Simulator 3-32, 15-62
SINEC L1 12-14
- local area network 12-7
Slot address assignment 6-1
Slot addressing 6-1
Slot coding 3-14
- element 3-13
Start-up 4-1
- PLC system 4-14
Starting
- timer 8-19
Statement list (STL) 7-1
STATUS 4-9
STATUS VAR 4-9
Status word
- structure 13-10
Step address counter 5-3
STL statement list
STOP 2-14
Structured programming 7-5
Suppression measures 3-41
Switch setting
- for distributed connection 3-23
SYNCHRON block 11-27
System
- components 1-2
- operation 8-65, A-15
- parameter 5-16
- time 2-22, 2-25
System data area 2-14
- address allocation 6-16

**T**

Thermocouple 10-7
Time
- base 8-16
- loading 8-16
Time-controlled program execution 7-20
Timer 2-4
- address allocation 6-18
- starting 8-19
- operation 8-15
Transducer 10-4, 10-11, 10-20, 10-21, 10-24, 10-29
Transfer operation 8-10
Transferring
- program 4-3
- register contents 8-67

**U**

User time 2-22

**W**

Watchdog module 15-60
Wirebreak signal 10-51
- in conjunction with resistance thermometers 10-52
Wiring 3-29
- arrangement 3-35

Siemens AG
A&D AS E 48
P.O. Box 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:

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____