

SIMATIC S5

IP 265

High Speed Sub Control

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

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.

Copyright © **Siemens AG 1993**

Preface

Introduction

System Overview

1

Technical Description

2

Installation Guidelines

3

General Operation

4

Addressing

5

Control of the IP 265 by the CPU User Program

6

Startup, Loading and Operating States

7

Fundamentals of COM 265

8

Programming the IP 265 with COM 265

9

Testing and Simulation with COM 265

10

COM 265 Services and File Functions

11

IP 265 Expansion

12

Sample Applications and Programming Aids

13

Standard Function "Counter"

14

Appendices

**A/B/C/
D/E
F/G**

Index

Summary

	Page
Preface	xi
Introduction	xiii
1 System Overview	1 - 1
1.1 System Environment of the IP 265	1 - 2
1.2 User Controls	1 - 3
2 Technical Description	2 - 1
2.1 Hardware Structure of the IP 265	2 - 1
2.2 Interface to the External I/O Bus	2 - 2
2.3 Interface to the Process I/O	2 - 3
2.3.1 24 V Digital Inputs	2 - 3
2.3.2 24 V Digital Outputs	2 - 3
2.3.3 5 V Differential Inputs	2 - 4
2.4 Expansion Interface	2 - 5
2.5 Connector for Load Voltage	2 - 5
2.6 Memory Submodule Receptacle	2 - 6
2.7 LEDs	2 - 6
3 Installation Guidelines	3 - 1
3.1 Assembly and Removal of the IP 265	3 - 1
3.2 Establishing Connection to the I/Os	3 - 1
3.2.1 Connecting 24 V Digital Inputs	3 - 2
3.2.2 Connecting 24 V Digital Outputs	3 - 2
3.2.3 Connecting Differential Inputs	3 - 3
3.2.4 Connecting Expansion Inputs/Outputs	3 - 3
3.3 Power Supply of the IP 265	3 - 4
3.4 Important Information on the Configuration and Installation of the IP 265	3 - 5
3.5 Hardware Fault Detection During Connection Buildup	3 - 6

	Page
4	General Operation 4 - 1
4.1	Parallel Program Execution with IP 265 4 - 1
4.2	Response Time 4 - 4
4.2.1	Program Execution Time 4 - 5
4.2.2	Input delay 4 - 6
4.2.3	Output delay 4 - 6
4.3	Output and Input Data 4 - 7
4.3.1	Structure and Operation of the Control Word 4 - 8
4.3.2	Structure and Handling of the Status Word 4 - 9
4.3.3	Parameters of the IP 265 User Program 4 - 12
5	Addressing 5 - 1
5.1	Addressing in the IP 265 User Program (IP 265 Viewpoint) 5 - 2
5.1.1	Actual Addresses of the Inputs and Outputs of the IP 265 5 - 2
5.1.2	Actual Addresses of the Input and Output Parameters (IP 265 Viewpoint) 5 - 3
5.2	Addressing in the CPU User Program (CPU Viewpoint) 5 - 4
5.2.1	Addressing of the Control Word and the Status Word 5 - 5
5.2.2	Actual Addresses of the Input and Output Parameters (CPU Viewpoint) 5 - 6
5.3	Allocation of the Parameter Addresses in the IP 265 User Program and in the CPU User Program 5 - 8
6	Control of the IP 265 by the CPU User Program 6 - 1
6.1	Control of IP 265 Program Execution 6 - 1
6.2	Transfer Times for Data Exchange Between CPU and IP 265 6 - 3
6.3	CPU Program Section for Control of the IP 265 6 - 4
6.3.1	Structure of the CPU Program Section 6 - 5
6.3.2	Programming blocks of the CPU Program Part 6 - 6
7	Startup, Loading and Operating States 7 - 1
7.1	Function Model of a Startup 7 - 1
7.2	Loading 7 - 5
7.2.1	Loading the IP 265 by Means of COM 265 via the External I/O Bus 7 - 5
7.2.2	Loading the IP 265 from the Memory Submodule 7 - 7
7.3	Operating States 7 - 8
7.3.1	Possible Operating States 7 - 8
7.3.2	Useful Operating States for Normal Operation and Transitions 7 - 9

	Page
8	Fundamentals of COM 265 8 - 1
8.1	Preparations for Working with COM 265 8 - 2
8.2	COM 265 Screen Forms 8 - 4
8.3	Basic COM 265 Functions in the "Function Selection" Form 8 - 5
8.4	Hierarchical Structure of COM 265 8 - 8
8.5	COM 265 Operator Control Philosophy 8 - 10
8.5.1	Help Forms and Help Windows 8 - 10
8.5.2	Errors and Warnings 8 - 13
8.5.3	Cursor Control in Control System Flowcharts and Input Fields 8 - 15
8.5.4	Key Assignments for Editing Functions 8 - 16
9	Programming the IP 265 with COM 265 9 - 1
9.1	Defaults 9 - 1
9.2	Configuring the IP 265 Response 9 - 4
9.3	COM 265 Language Description 9 - 6
9.3.1	Operands in a Control System Flowchart 9 - 8
9.3.2	Language Elements in a Control System Flowchart 9 - 11
9.4	Enter IP 265 User Program 9 - 27
9.4.1	Rules and Recommendations for Programming 9 - 27
9.4.2	Entries in the Local IP 265 Assignment List 9 - 30
9.4.3	Entering a Control System Flowchart 9 - 33
9.4.4	Entering the Segment Name and Segment Commentary 9 - 46
9.5	Compiling the IP 265 User Program 9 - 48
9.5.1	IP 265 User Program Load on the IP 265 9 - 49
9.5.2	What if there is an IP 265 Overload? 9 - 51
10	Testing and Simulation with COM 265 10 - 1
10.1	Off-Line Simulation of the IP 265 User Program 10 - 1
10.1.1	General Information on Simulation with COM 265 10 - 1
10.1.2	Starting the Simulator and Simple Simulation of a COM 265 Language Element 10 - 4
10.1.3	Generating Simulator Settings 10 - 6
10.1.4	Resetting Simulator Settings 10 - 14
10.1.5	Symbols for Simulator Settings 10 - 15
10.1.6	Clocked Simulator Control 10 - 17
10.1.7	Text Representation 10 - 18

	Page
10.2 On-Line Testing of the IP 265	10 - 20
10.3 Wiring Test	10 - 23
11 COM 265 Services and File Functions	11 - 1
11.1 Services	11 - 1
11.1.1 Invoking the Symbols Editor	11 - 2
11.1.2 Loading the IP 265 Via the I/O Bus	11 - 2
11.1.3 Storing an IP 265 User Program on a Memory Submodule	11 - 4
11.1.4 Reading an IP 265 User Program from a Memory Submodule	11 - 6
11.1.5 Printing an IP 265 User Program	11 - 8
11.2 File Functions	11 - 9
11.2.1 Copying an IP 265 User Program	11 - 10
11.2.2 Deleting an IP 265 User Program	11 - 11
11.2.3 Display Directory of All IP 265 User Programs	11 - 12
11.2.4 Renaming an IP 265 User Program	11 - 13
12 IP 265 Expansion	12 - 1
12.1 General Remarks	12 - 1
12.2 Interconnecting Two IP 265s	12 - 5
12.3 Power Supply for Expanded IP 265s	12 - 5
12.4 Addressing the Expansion Inputs/Outputs	12 - 6
12.5 Sample Program	12 - 6
13 Sample Applications and Programming Aids	13 - 1
13.1 Sample Programs	13 - 1
13.1.1 Simple Sample Program	13 - 2
13.1.2 Sample Program with Parameter Interchange	13 - 7
13.1.3 Sample Program with Expansion and Parameter Interchange	13 - 13
13.2 Programming Aids	13 - 14
14 Standard Function "Counter"	14 - 1

Page

Appendices

A	Diagnostics and Error Messages	A - 1
	A.1 LEDs	A - 1
	A.2 IP 265 Error Messages	A - 2
B	Technical Specifications	B - 1
C	Dimension Drawing of the IP 265	C - 1
D	Keyboard Layout for COM 265 Editing Functions on the Programmer	D - 1
E	Glossary	E - 1
F	Active and Passive Faults in Automation Systems	F - 1
G	Accessories and Order Numbers	G - 1

Index

Preface

The IP 265 represents an innovative enhancement of the processing power of the PLC (PLC=Programmable Logic Control). It is a powerful, **freely programmable** I/O module and a member of the range of I/O modules common to the S5-100U, S5-90U/S5-95U and ET 200U (distributed I/O system).

On the worldwide market of mini and micro PLCs, there is a growing tendency towards faster and smaller components. In the case of existing mini PLCs of the SIMATIC S5 family, system-dependent limits are set by processing speed of control commands. The conventional SIMATIC C1, C2, C3 circuitry is therefore still used for especially fast applications. With this technology, hardware connections which enable high-speed sub control but are not reversible in the sense of "programmable" are set up by means of electrical wiring (solder bonds).

The use of **FPGA technology** in the IP 265 makes it possible for the first time to meet high speed requirements for **sub controls** of an overall system with a freely programmable SIMATIC S5 module. Hardware connections are also set up on the IP 265 by loading memory contents into an FPGA (**Field Programmable Gate Array**). The hardware connections established in this way enable **parallel** and therefore **fast** sub control of an overall system. The FPGA used in the IP 265 also has the advantage that the sub control can easily be modified by renewed loading of FPGA-relevant data.

A special user program (**IP 265 user program**) must be loaded into the IP 265. There are two options:

- A program generated to suit specific user requirements or
- Fixed-programmed standard software from Siemens

A special COM (**COM 265**) is available for free programming of the IP 265.

In addition to the programmable application, the IP 265 also offers the possibility of implementing the special "timer" function by means of a standard program. Siemens offers a memory submodule with the standard function counter for the IP 265. It is supplied separately and is not described in this manual.

The program capacity of the FPGA used in the IP 265 is limited. The IP 265 is therefore used for high-speed sub controls. More complex sub processes can be controlled with the IP 265 by **adding** a further IP 265 module.

Note

In the manual, Section 13 "Sample Applications and Programming Aids" is reserved for program examples. It enables fast startup of the module without having to read the entire manual.

Introduction

Before you continue, you should read the introduction carefully. This facilitates use of the manual and saves time.

To use the IP 265 optimally, you need detailed information.

In the manual (GHB), we have attempted to present this information as completely and as well organized as possible. At the same time, we have attempted to meet the increased demands placed on technical documentation. This means in particular:

- Standardization of the vocabulary and notation
- Extensive organization
- Visualization of individual topics
- Customer-oriented layout of the contents

In this way we would like to provide the information necessary for working with the IP 265 to both inexperienced as well as experienced SIMATIC S5 users.

Description of contents

This manual represents a comprehensive description of the IP 265. It can be divided into individual thematic blocks.

- Description (Sections 1 and 2)
 - The "System Overview" informs you of the functions the module can handle and how it is used in the S5-100U, S5-90U/95U and ET 200.
 - The "Technical Description" provides general information on the interfaces of the module, information on the pin assignments and technical specifications.
- Assembly and connection buildup (Section 3)
 - The "Installation Guidelines" provide you with all the information you need to assemble the module and link it with the process I/O. Furthermore, you are informed of the voltages the IP 265 must be supplied with during normal operation.
- Mode of operation (Sections 4 to 7)
 - We describe here generally how the module works in its system environment, giving you information on parallel program execution and addressing of the IP 265 as well as control of IP 265 program execution by the CPU. Using a function model, we will demonstrate the basic user actions required for starting up the IP 265.
- Working with COM 265 (Sections 8 to 11)
 - These sections give a detailed description of the necessary user inputs during startup of the IP 265 with the aid of COM 265. It shows how the module functions are defined, simulated off-line and tested on-line.
- Expansion (Section 12)

The expansion of two IP 265 modules is a special case. Module expansion is dealt with in a separate section of this manual.

- Programming examples (Sections 12 and 13)
In the last sections of the manual, you will find several application examples and programming aids. They should illustrate the diverse possible applications of the IP 265 and enable fast startup of the module.
- Standard programs (Section 14)
At this point in the manual, a placeholder is inserted for separately available documentation on the standard function counter which can be implemented in the IP 265.
- Overviews (appendices)
We have arranged specifications in a clear manner to simplify use of the IP 265. You will find a complete list of all error messages of the IP 265, a summary of the technical specifications of the IP 265 (including dimension drawing), guidelines for maintenance and service and an accessories list with order numbers. There is a glossary containing all the terms which we had to define in conjunction with the application of a new technology in a SIMATIC S5 module.

At the end of the manual, correction forms are included. Please enter in these forms any suggestions you may have for improvements, additions or corrections and send them to us. They will help us to improve the next edition of this manual.

Conventions

This manual is organized in menu form to make it easier for you to find information. This means the following:

- Each section is marked with printed tabs.
- On the next two pages of the manual, you will find an overview page that lists the title of each section.
- The overview page is followed by the detailed table of contents.
- The detailed table of contents in turn precedes every section.
Each section has three level headings that are numbered. The fourth level heading is not numbered but appears in **boldface type**.
- Pages, figures and tables are numbered separately for each section. On the back of the detailed table of contents for each section, you will find a list of the figures and tables that appear in that section.

The manual employs the following specific structuring devices:

- Specific terms have characteristic abbreviations.
Example: programmer (PG)
- Footnotes are marked with a raised asterisk "*". You will find the corresponding explanations in the lower margin of the page or below the table.
- Lists are designated with bullets (•) (as in this particular listing) or dashes (-), or they are numbered consecutively ().
Operator actions are indicated by a black triangle ().
- Cross-references are indicated as follows:
"(Section 7.3.2)".
There are no references to specific page numbers.
- Dimensions in drawings are indicated in "mm".
- Value ranges are indicated as follows: 17 to 21
- Especially important information appears in framed boxes such as the following:

<p>Note</p> <hr/>

You will find definitions of the terms "Warning" and "Note" in the next section, "Safety-Related Guidelines for the User".

Safety-Related Guidelines for the User

This document provides the information required for the intended use of the IP 265. 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 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.

Warning

indicates that death, severe personal injury or substantial 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.

Courses

Siemens offers extensive opportunities for training to SIMATIC S5 users. For detailed information, please contact your local Siemens office.

1	System Overview	
1.1	System Environment of the IP 265	1 - 2
1.2	User Controls	1 - 3

Figures

1-1	System Environment of the Module	1 - 2
-----	--	-------

1 System Overview

Intelligent I/O modules extend the field of application of SIMATIC S5 programmable controllers. They are technology-oriented and offload the central processor in the CPU.

The IP 265 is an extremely powerful I/O module. Use of the module in an S5 system permits fast I/O preprocessing through parallel program scanning.

For complex applications, only one fast sub process of an overall system is controlled by the IP 265. For this purpose, the sub process must be transferred from the CPU (CPU user program) to the IP 265 (IP 265 user program).

IP 265 has the following performance characteristics:

- Recording of fast processes (e.g. counting)
- Control of fast processes (e.g. control of stepper motors)
- Short response times (e.g. fast disconnection of actuators)
- Data exchange with CPU user program (e.g. assignment of sub control parameters (IP 265))

The module can be used with the CPUs of the following S5 systems:

- S5-100U (CPU 100 from 6ES5 100-8MA02, CPU 102 from 6ES5 102-8MA02 or CPU 103)
- S5-90U/95U
- ET 200U (IM 318-B from version 2)
Restriction: The IP 265 can
 - only be loaded from memory submodule,
 - only be operated in Slow mode in the ET 200U.

The S5 system forms the environment of the IP 265 during operation of the module.

Programming and startup of the IP 265 are carried out by means of COM 265 at the programmer. In principle, COM 265 can also be loaded on a PC.

Convention

All S5 systems with which the IP 265 can be used are commonly denoted as "S5 100" in this manual.

1.1 System Environment of the IP 265

Use of the IP 265 enables the control of fast sub processes of an overall system. For communication with the system environment, the module is equipped with several ports. The SIMATIC S5 hardware constitutes the direct environment of the IP 265.

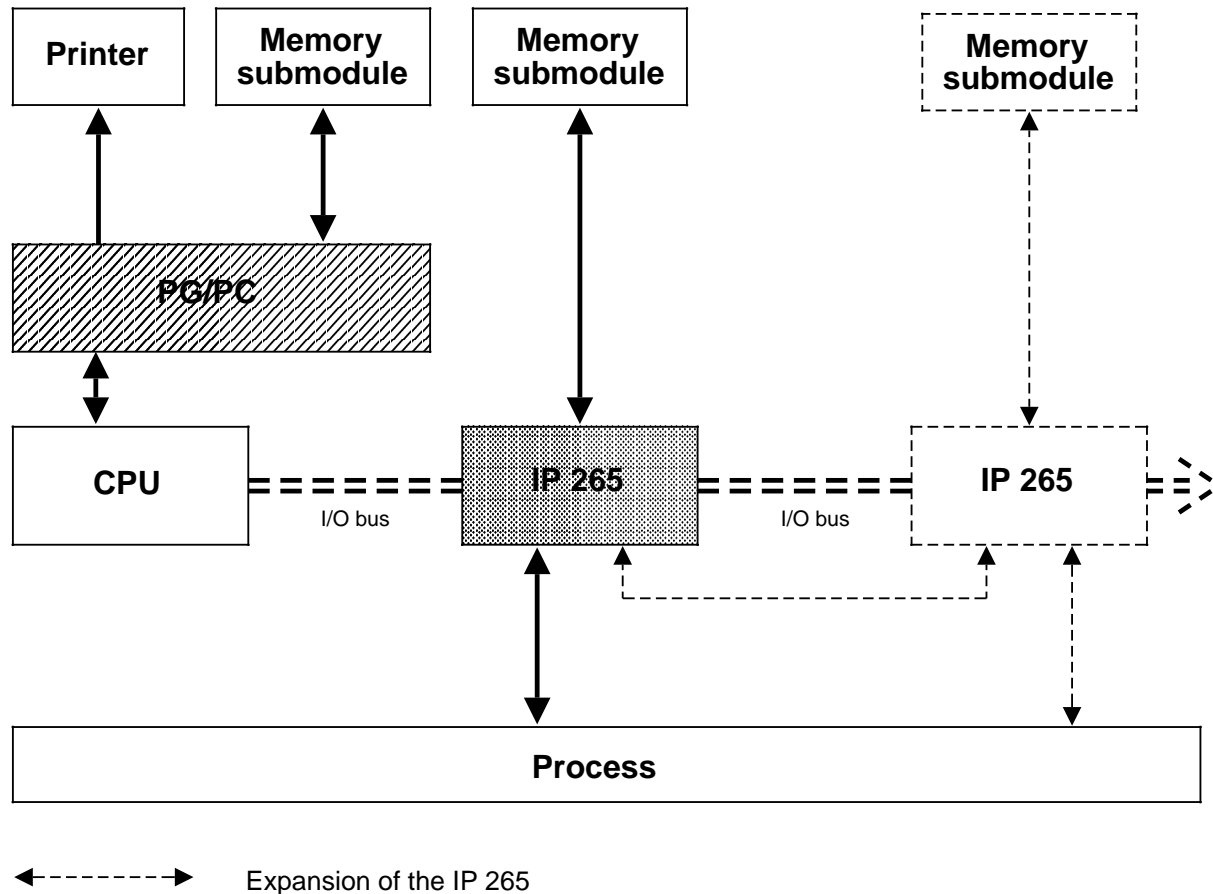


Figure 1-1. System Environment of the Module

Interface to the external I/O bus

In normal operation, the IP 265 (and the sub process controlled by it) communicates with the CPU user program.

- The CPU user program transfers control commands to the IP 265 via the I/O bus and evaluates their statuses.
- Via the I/O bus, parameters can additionally be exchanged between the CPU user program and the IP 265 user program.

Interface to the process

The fast sub process is processed independently by the IP 265. The inputs and outputs of the module communicating with the process are local and cannot be accessed direct via the CPU.

Submodule receptacle for accommodating a memory submodule

The memory submodule is required for storage of the IP 265 program data.

Interface to another IP 265

The results obtained in one IP 265 can be processed by the other IP 265.

The expansion enables you to

- nearly double the program capacity,
- double the number of inputs and outputs.

You can define the IP 265 function easily at the programmer/PC.

- Generation of the IP 265 user program
- Generation of a program section of the CPU user program for control of IP 265 program execution.

The printer is used for documenting the IP 265 user program. You can print out the complete user program including a symbols list.

1.2 User Controls

Only with the programmer/PC can you influence the IP 265 and its functions direct.

Please bear in mind:

The control of the overall system must be shared by the CPU user program and the IP 265 user program. You can additionally determine the control of IP 265 program execution in the CPU user program.

Serial COM software is available for starting up the IP 265. **COM 265** enables user-friendly programming and function testing of the IP 265 for your special application.

COM 265 is incorporated in the STEP 5 environment of the programmer. However, COM 265 is not invoked like conventional SIMATIC-COM packages via the STEP 5 command interpreter on the basis of the PCP/M-86 operating system. COM 265 is supported by the MS-DOS operating system and can only be operated in connection with a specific S5 emulator for programmers (Section 8.1). COM 265 runs on the PG 730, PG 750 and PG 770 programmers.

Generally, COM 265 can also be used on a PC. In Section 8.1 and in Appendix D of the manual, you will find all the necessary information on operating COM 265 on a PC.

The CPU is programmed by means of the **STEP 5 STL/LAD/CSF** package (invoked via STEP 5 command interpreter). Since the IP 265 can only be used together with the S5-100U, S5-90U/95U and the ET 200U, we assume that you have a manual for one of these systems available. The basic knowledge provided there is sufficient for starting up the IP 265 together with the CPU.

Note

If you use a standard program to define the function of the IP 265, you do not require COM 265. The standard program cannot be changed by means of COM 265.

2 Technical Description		
2.1	Hardware Structure of the IP 265	2 - 1
2.2	Interface to the External I/O Bus	2 - 2
2.3	Interface to the Process I/O	2 - 3
2.3.1	24 V Digital Inputs	2 - 3
2.3.2	24 V Digital Outputs	2 - 3
2.3.3	5 V Differential Inputs	2 - 4
2.4	Expansion Interface	2 - 5
2.5	Connector for Load Voltage	2 - 5
2.6	Memory Submodule Receptacle	2 - 6
2.7	LEDs	2 - 6

Figures		
2-1	Block Diagram of the IP 265 with Ports	2 - 1
2-2	Overview of the Module	2 - 2
2-3	Pin Assignments of the Interface for the 24 V Inputs	2 - 3
2-4	Pin Assignments of the Interface for the 24 V Outputs	2 - 3
2-5	Pin Assignments of the Interface for the 5 V Differential Inputs	2 - 4
Tables		
2-1	Storage Possibilities for Memory Submodules	2 - 6

2 Technical Description

2.1 Hardware Structure of the IP 265

The IP 265 is a single-width module of the S5-100 system. The module is equipped with several ports for communication with the system environment.

The following representation shows an overview block diagram in which the essential hardware components of the module are shown.

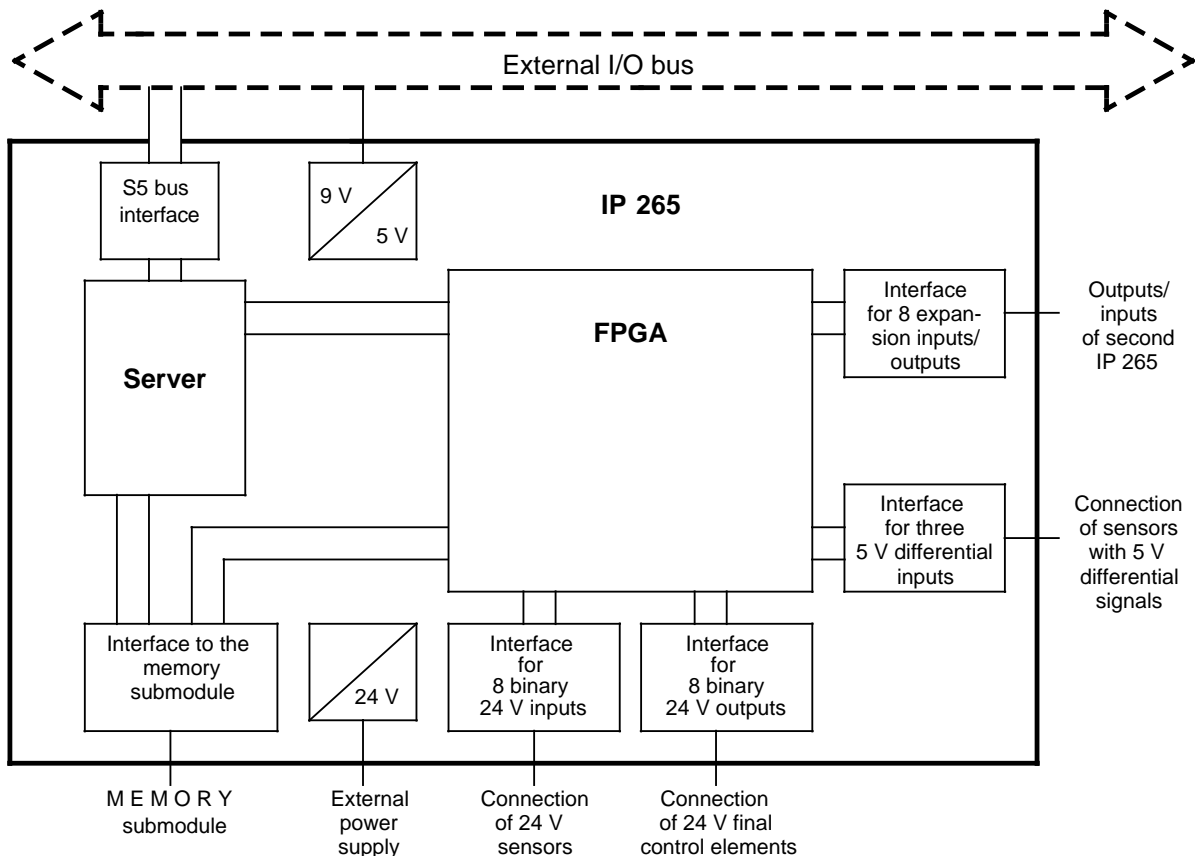


Figure 2-1. Block Diagram of the IP 265 with Ports

FPGA (Field Programmable Gate Array):

On the FPGA, hardware structures are generated during startup of the IP 265. For this purpose, an IP 265 user program must be generated and loaded into the module. In the IP 265 user program, you determine the sub control of your system.

Server:

The server handles communication between the FPGA and the CPU. The server additionally monitors the IP 265 and outputs status reports of the IP 265 to the I/O bus.

Interfaces:

Via the interfaces, the IP 265 can communicate

- with the process
- with another IP 265
- with the CPU

Figure 2-2 shows the position of the interfaces on the front of the module.

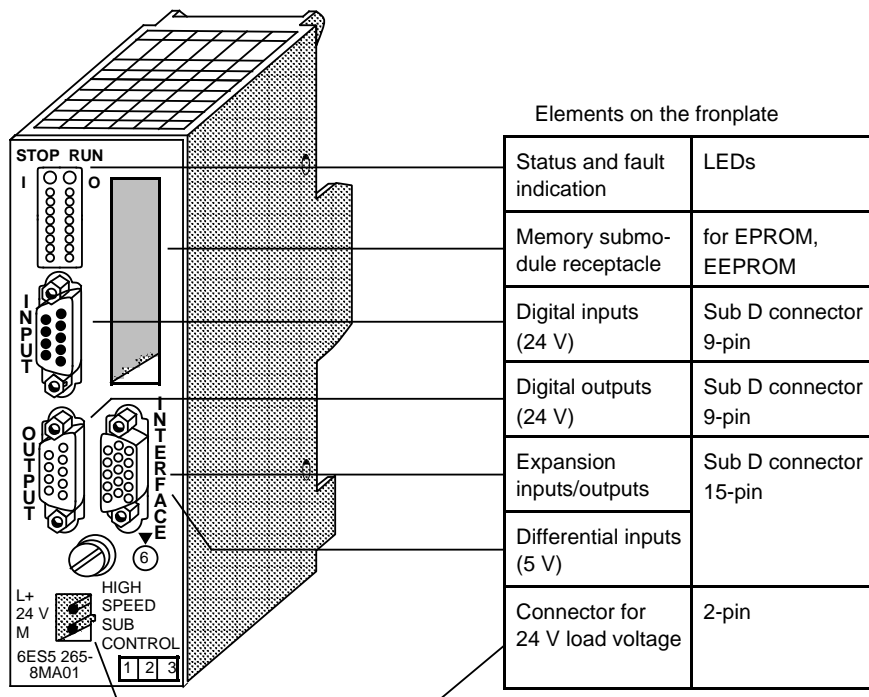


Figure 2-2. Overview of the Module

Note

In order to avoid confusion of the connections of the 24 V digital inputs and 24 V digital outputs, a 9-pin sub D **plug connector** and/or a 9-pin sub D **socket connector** is available on the module.

Note

You will find a summary of all technical specifications of the module in Appendix B of the manual: "Technical Specifications".

2.2 Interface to the External I/O Bus

The interface to the external I/O bus is on the rear of the module. Section 3.1 "Assembly and Removal of the IP 265" tells you how to connect the IP 265 and the I/O bus.

2.3 Interface to the Process I/O

2.3.1 24 V Digital Inputs

Connect the signal lines of the plant sensors to the 9-pin sub D connector labelled "INPUT". You can connect up to 8 sensors with 24 V signals. The frequency of the input signals may be at the most 10 kHz. For connecting sensors with chattering contacts, you can configure an input delay in the range of 1 to 2 ms for every input.

For the pin assignment of the 9-pin sub D connector refer to the following diagram.

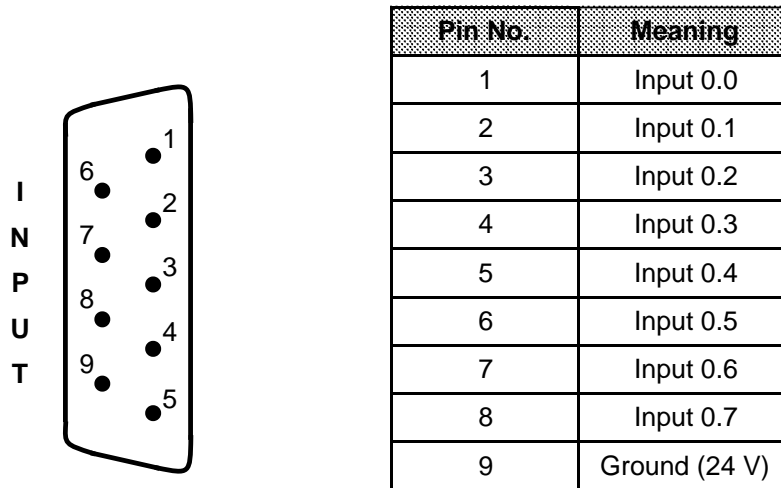


Figure 2-3. Pin Assignments of the Interface for the 24 V Inputs

2.3.2 24 V Digital Outputs

You connect the signal cables of the actuators in the plant to the 9-pin sub D socket labelled "OUTPUT". You can connect up to 8 actuators with 24 V signals. The response time of the 24 V outputs is dependent on the resistive load of the output circuit. It is typically 70 μ s for a maximum resistive load.

You see the pin assignments of the 9-pin sub D socket in the following diagram.

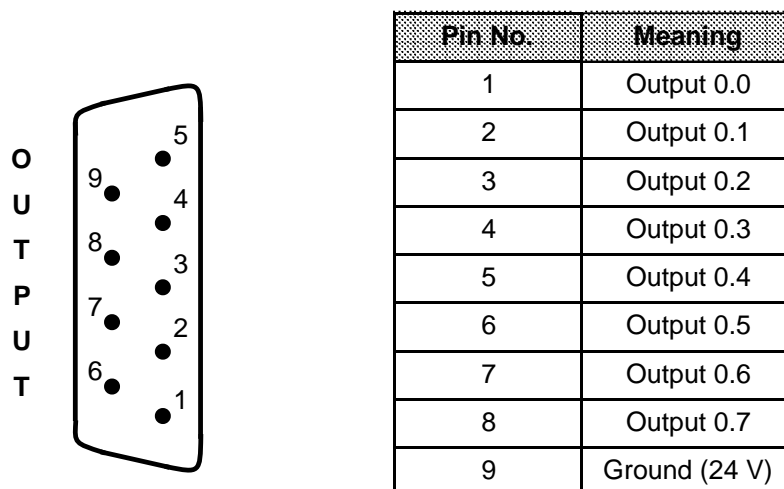


Figure 2-4. Pin Assignments of the Interface for the 24 V Outputs



Warning

Parallel connection of the 24 V outputs is only allowed in pairs. The output current per 24 V output is thus 0.8 times the rated current.

2.3.3 5 V Differential Inputs

The IP 265 has 3 differential inputs. The differential inputs are connected to the 15-pin sub D socket of the module. The signal levels at the interface of the differential inputs correspond to the RS 422 standard.

You can connect 5 V sensors with differential signals to the differential inputs. The maximum input frequency is 58 kHz. The pulse length of the input signals at the 5 V differential inputs (signal "0" (0 V) or "1" (5 V)) must be at least 8.6 μs.

Unwired ("open") 5 V differential inputs are handled by the IP 265 as differential inputs with signal state "1" (5 V).

The following diagram shows the assignments for the signal cables of the 5 V sensors on the 15-pin sub D socket. 6 pins are reserved for the 3 differential inputs on the sub D socket. The remaining 8 pins of the 15-pin sub D socket are irrelevant for the use of the differential inputs (Section 2.4).

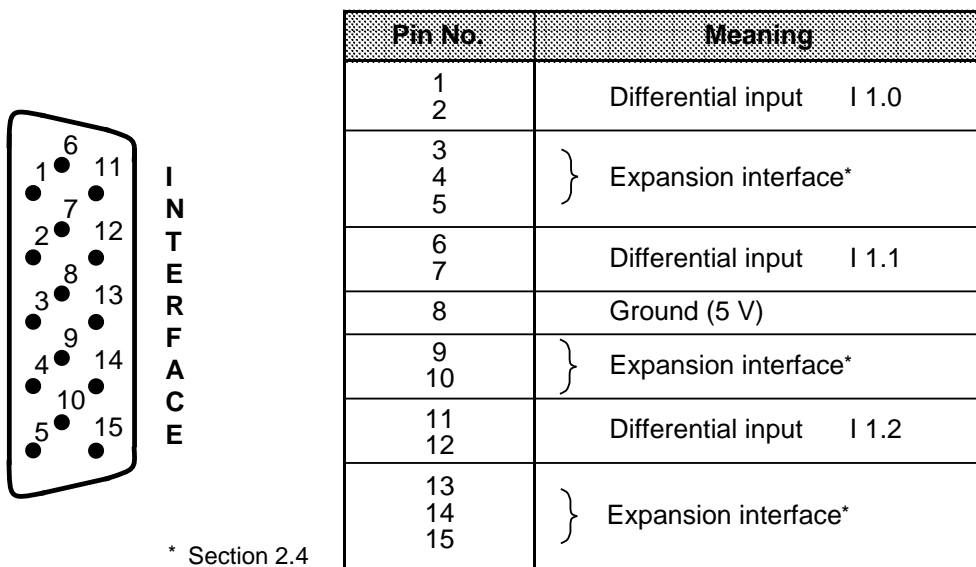


Figure 2-5. Pin Assignments of the Interface for the 5 V Differential Inputs

Note

The differential inputs are connected to the same 15-pin sub D socket as the expansion interface (Section 2.4). When using the differential inputs, simultaneous use of the expansion interface is not possible.

2.4 Expansion Interface

You can link the IP 265 with a further IP 265 via the expansion interface. This is only possible with the special connection cable offered by Siemens.

This extension can become necessary if

- the number of digital inputs and outputs of **one** IP 265 (24 V) or
- the program capacity of **one** IP 265

are not sufficient in the case of a more complex application.

The 15-pin sub D socket of the module has 8 pins for expansion purposes. The pins of the expansion interface are optionally available (freely configurable with COM 265) as expansion inputs or as expansion outputs. The remaining 6 pins of the sub D socket are irrelevant (Section 2.3.3) for the expansion.

Note

The expansion inputs/outputs are connected to the same sub D socket as the differential inputs (Section 2.3.3). When using the expansion inputs/outputs, simultaneous use of the differential inputs is not possible.

Unwired ("open") expansion inputs/outputs are handled by the IP 265 as expansion inputs with signal state "1" (5 V).



Warning

Unlike the 24 V outputs, the extension outputs are not reset in the case of STOP of the IP 265. They maintain their original signal state "0" (0 V) or "1" (5 V).
The extension inputs/outputs are to be used exclusively for expansion purposes. No sensors or actuators may be connected to the expansion inputs/outputs.

2.5 Connector for Load Voltage

The load circuit of the system control is supplied with 24 V DC via the load voltage connector.

Note

Only if the 24 V DC load voltage is connected to the IP 265 can the module be switched to RUN.

2.6 Memory Submodule Receptacle

The submodule receptacle of the IP 265 can accommodate an EPROM or EEPROM submodule.

Generally, the user program for the IP 265 is to be stored on the memory submodule. If you use a standard program, you can also plug the fixed-programmed memory submodule supplied (EPROM) into the submodule receptacle of the IP 265.

The memory submodules permissible for the IP 265 are listed in Appendix G of the manual: "Accessories and Order Numbers".

Table 2-1. Storage Possibilities for Memory Submodules

Submodule Type	Store IP 265 Program with
EPROM	Programmer
EEPROM	Programmer/IP 265



Warning

The memory submodule may only be inserted or withdrawn in the POWER OFF state of the CPU.

2.7 LEDs

The frontplate of the IP 265 is equipped with several light-emitting diodes:

- 1 red STOP LED
- 1 green RUN LED
- 8 green LEDs for 8 24 V digital inputs
- 8 green LEDs for 8 24 V digital outputs

The operating state of the IP 265 is indicated by either a bright RUN LED or STOP LED. Errors in the IP 265 are indicated via the STOP LED and/or a combination of the RUN and STOP LEDs. Various errors are indicated by different flashing signals. You will find a summary of all possible status indications of the IP 265 in Appendix A of the manual: "Diagnostics and Error Messages".

The LED matrix (16 LEDs arranged in pairs) is used to indicate the signal state of each 24 V input and 24 V output.

3 Installation Guidelines

3.1	Assembly and Removal of the IP 265	3 - 1
3.2	Establishing Connection to the I/Os	3 - 1
3.2.1	Connecting 24 V Digital Inputs	3 - 2
3.2.2	Connecting 24 V Digital Outputs	3 - 2
3.2.3	Connecting Differential Inputs	3 - 3
3.2.4	Connecting Expansion Inputs/Outputs	3 - 3
3.3	Power Supply of the IP 265	3 - 4
3.4	Important Information on the Configuration and Installation of the IP 265	3 - 5
3.5	Hardware Fault Detection During Connection Buildup	3 - 6

Figures		
3-1	Typical Connection for 24 V Digital Inputs	3 - 2
3-2	Typical Connection for 24 V Digital Outputs	3 - 2
3-3	Typical Connection for 5 V Differential Inputs	3 - 3
3-4	Simplified Representation of Nonfloating Connection of the IP 265 to the PLC	3 - 4
Tables		
3-1	Indication and Effects of Error Conditions	3 - 6

3 Installation Guidelines

3.1 Assembly and Removal of the IP 265

The IP 265 is to be snapped onto a bus unit in the same way as an I/O module of the S5-100 system. Please note the following for assembly and removal of the module:

- The IP 265 may only be connected or disconnected without load voltage and in the STOP mode of the CPU.
- The memory submodule may only be inserted or removed in the POWER OFF state of the CPU.
- The IP 265 may only be plugged into slots 0 to 7.
- The maximum permissible number of IP 265 modules is limited depending on the S5 system:
 - In the S5-100, S5-95 and ET 200, a maximum of six IP 265 modules can be connected.
 - In the S5-90, you can connect up to two IP 265 modules

Setting the coding element

Like every I/O module, the IP 265 is provided with a white coding key on the back. The coding key of the IP 265 is always set to number 6. The bus unit has a white, rotating counterpart for every slot: the coding element. If you want to snap the module onto the bus unit, you must also turn the coding element to "6" so that the coding key can snap into place in the coding element.

3.2 Establishing Connection to the I/Os

The following electrical connections are to be established according to your particular application:

- Connection of 24 V digital inputs of the module to sensors in the plant
- Connection of 24 V digital outputs of the module to actuators in the plant
- Connection of differential inputs of the module to encoders with error signals
- Connection of expansion inputs/outputs of two IP 265 modules.

Sections 3.2.1 to 3.2.4 show you how to establish the connections.

Note

The shields of the signal cables are to be connected to the connector casing and the shield/protective ground conductor rail.

Cable interface conditions

For the correct operation of the IP 265, important principles must be considered when **laying** and **shielding** the connecting cables. Since the IP 265 can only be used in connection with the S5-100, S5-90/95 or ET 200, we assume that you have one of the relevant manuals available. These manuals contain detailed explanations of all cable interface conditions for programmable controllers (PLCs).

3.2.1 Connecting 24 V Digital Inputs

The signal cables of sensors in the plant are to be connected via the 9-pin sub D connector of the IP 265 labelled "INPUT". The digital inputs of the IP 265 are designed for 24 V DC and must be non-floating. For the power supply of the input circuit, 24 V DC must be supplied via the load voltage connector on the front of the module (Section 3.3).

The maximum length of the signal cables is 100 m. The signal cables must be shielded.

Figure 3-1 shows two connection examples for the 24 V inputs of the IP 265.

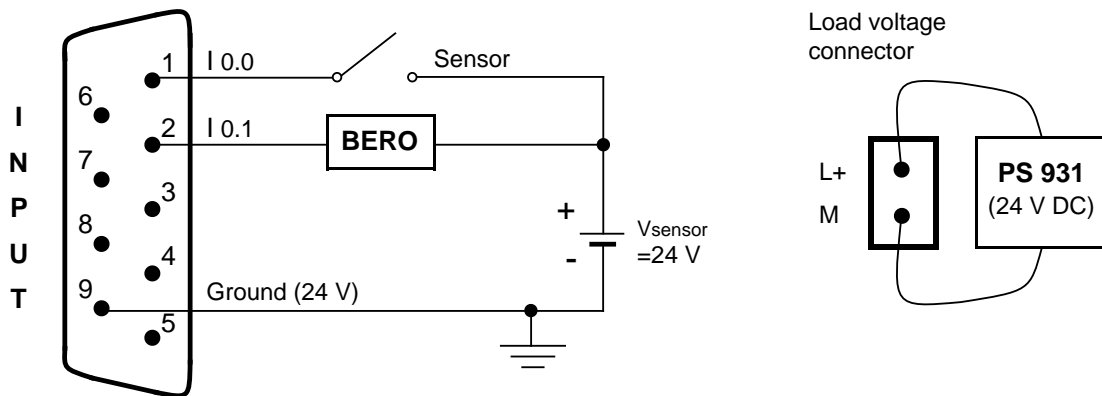


Figure 3-1. Typical Connection for 24 V Digital Inputs

3.2.2 Connecting 24 V Digital Outputs

The signal cables to the actuators in the plant are to be connected to the 9-pin sub D socket "OUTPUT". The digital outputs of the IP 265 are to be designed for 24 V DC and must be non-floating. For the power supply of the output circuit, 24 V DC must be supplied to the load voltage connector on the front of the module (Section 3.3).

Signal cables with a maximum length of 100 m can be used. Parallel connection of the 24 V outputs is only allowed **in pairs**. The output current per 24 V output is thus 0.8 times the rated current.

Figure 3-2 shows a connection example for the 24 V outputs of the IP 265.

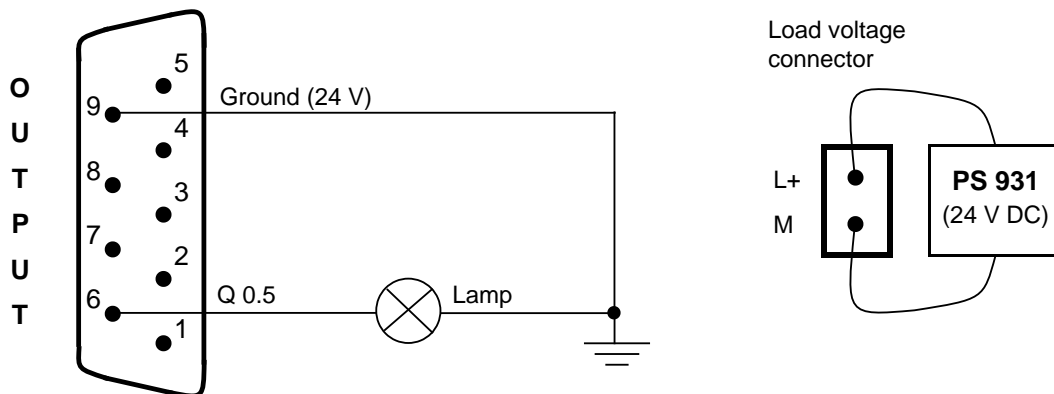


Figure 3-2. Typical Connection for 24 V Digital Outputs

3.2.3 Connecting Differential Inputs

The signal cables of encoders with error signals are to be connected to the 15-pin sub D socket of the IP 265 labelled "INTERFACE". The 3 differential inputs of the IP 265 are designed for 5 V. The maximum length of the signal cables is 32 m. The signal cables **must** be shielded. Prefabricated standard cables from Siemens may be used for this purpose (Appendix G). Figure 3-3 shows a connection example for the 5 V differential inputs of the IP 265.

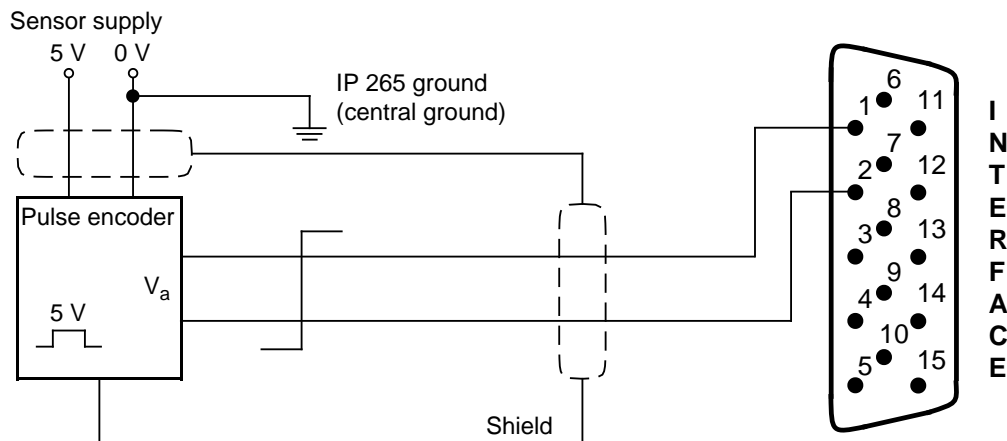


Figure 3-3. Typical Connection for 5 V Differential Inputs

Note

Do not use prefabricated cables, and if a sensor with 5 V error signals is to be connected to the IP 265, the 5 V sensor supply cable is also to be shielded (Figure 3-3).

3.2.4 Connecting Expansion Inputs/Outputs

For the electrical connection of two IP 265 modules, a prefabricated, shielded standard cable is available (expansion cable). Use the expansion cable to link the two 15-pin sub D sockets marked with "INTERFACE".

Note

The expansion inputs/outputs are to be connected exclusively via the expansion cable especially provided for this purpose (Appendix G).

The expansion inputs/outputs are connected to the 15-pin sub D socket labelled "INTERFACE" together with the differential inputs. When using the expansion inputs/outputs, the 5 V differential inputs cannot be used.

3.3 Power Supply of the IP 265

The IP 265 must be connected to a power supply for operation:

24 V DC supply

Load voltage is supplied via the two-pole connector on the front panel of the IP 265. The 24 V DC external supply enables the connection of 24 V sensors and actuators in the plant to the digital inputs and outputs of the module (load circuits).

You can connect the following:

- PS 931 power supply module (Appendix G)
or
- Siemens load power supply of the 6EW1 series.

When connecting other load power supplies, please note that the voltage must be in the range of 20 to 30 V (including ripple).

The IP 265 is a non-floating module. The 24 V DC load circuits and the control circuit of the PLC have a common reference potential.

Figure 3-4 shows the simplified connection of the IP 265 to the PLC.

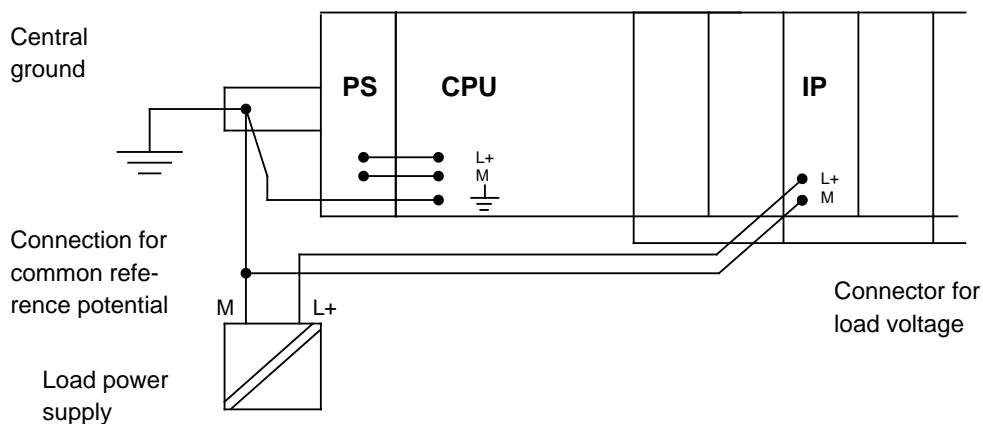


Figure 3-4. Simplified Representation of Nonfloating Connection of the IP 265 to the PLC

Note

When using the IP 265, you have to connect the ground potential of the IP 265 load voltage connector to the ground of the CPU via an external connection at central ground (Figure 3-4). Please ensure that the connecting cable has a low impedance (must be as short as possible).

3.4 Important Information on the Configuration and Installation of the IP 265

Since the module usually constitutes a part of a larger system or plant, this information is intended as a guideline for the safe integration of the product into its environment.

Guidelines to be considered for installation and startup of the product - depending on the application:



Warning

- Follow strictly the safety and accident prevention rules that apply in each particular case.
- In the case of equipment with a permanent power connection which is not provided with an isolating switch and/or fuses which disconnect all poles, a suitable isolating switch or fuses must be provided in the building wiring system (distribution board). Furthermore, the equipment must be connected to a protective ground (PE) conductor.
- Before switching on the equipment, make sure that the voltage range setting on the equipment corresponds to the local power system voltage.
- In the case of equipment operating on 24 V DC, make sure that proper electrical isolation is provided between the mains supply and the 24 V supply. Only use power supply units to IEC 364-4-41 or HD 384.04.41 (VDE 0100 Part 410).
- Fluctuations or deviations of the power supply voltage from the rated value should not exceed the tolerances specified in the technical specifications. Otherwise, functional failures or dangerous conditions can occur in the electronic modules/equipment.
- Suitable measures must be taken to make sure that programs that are interrupted by a voltage dip or power supply failure resume proper operation when the power supply is restored. Care must be taken to ensure that dangerous operating conditions do not occur even momentarily. If necessary, the equipment must be forced into the "emergency off" state.
- Emergency tripping devices in accordance with EN 60204/IEC 204 (VDE 0113) must be effective in all operating modes of the automation equipment. Resetting the emergency off device must not result in any uncontrolled or undefined restart of the equipment.
- Install the power supply and signal cables in such a manner as to prevent inductive and capacitive interference voltages from affecting the automation functions.
- Automation equipment and its operating elements must be installed in such a manner 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, suitable hardware and software measures must be taken when interfacing the inputs and outputs of the automation equipment.

3.5 Hardware Fault Detection During Connection Buildup

Hardware faults on the 24 V DC load circuit side are indicated as follows:

- By flashing LEDs on the IP 265 (Appendix A.1)
- By error bits set in the status word of the IP 265 (Appendix A.2)

Table 3-1. Indication and Effects of Error Conditions

Hardware Fault	Error Message	Effects
<ul style="list-style-type: none"> • Absence of 24 V DC power supply 	<ul style="list-style-type: none"> • STOP LED flashes • I/O error bit set in the status word 	<ul style="list-style-type: none"> • IP 265 goes into the STOP mode
<ul style="list-style-type: none"> • Short-circuit between a 24 V output and ground 	<ul style="list-style-type: none"> • STOP LED flashes • RUN LED: Steady light • I/O error bit set in the status word 	<ul style="list-style-type: none"> • IP 265 still in the RUN mode • Affected short-circuited 24 V outputs are disconnected and set again after elimination of the short-circuit
<ul style="list-style-type: none"> • Short-circuit between two outputs 	<ul style="list-style-type: none"> • None 	<ul style="list-style-type: none"> • None
<ul style="list-style-type: none"> • Ground conductor interrupted (L+wire of 24 V DC supply connected, M wire is interrupted or not connected) 	<ul style="list-style-type: none"> • STOP LED flashes • I/O error bit set in the status word • All 8 LEDs for 24 V outputs: Steady light 	<ul style="list-style-type: none"> • The IP 265 enters the STOP state • 24 V digital inputs are not read • 24 V digital outputs are reset



Warning

In the case of a break in the ground connector, voltages up to 7 V can be output at the 24 V digital outputs in the case of high-impedance loads.

4 General Operation		
4.1	Parallel Program Execution with IP 265	4 - 1
4.2	Response Time	4 - 4
4.2.1	Program Execution Time	4 - 5
4.2.2	Input delay	4 - 6
4.2.3	Output delay	4 - 6
4.3	Output and Input Data	4 - 7
4.3.1	Structure and Operation of the Control Word	4 - 8
4.3.2	Structure and Handling of the Status Word	4 - 9
4.3.3	Parameters of the IP 265 User Program	4 - 12

Figures

4-1	Comparison of Sequential Program Execution and Parallel Program Execution	4 - 1
4-2	Example: Binary Segment of the IP 265 User Program	4 - 3
4-3	Time Diagram - Response Time of IP 265	4 - 4
4-4	Examples: Program Execution Times in Parallel Paths of a Segment	4 - 5
4-5	Assignment of the PIQ and PII in the CPU	4 - 7
4-6	Structure of the Control Word	4 - 8
4-7	Structure of the Status Word	4 - 9
4-8	Error Messages and Status Information in the Status Word	4 - 10

Tables

4-1	Essential Differences Between Sequential and Parallel Program Processing	4 - 3
4-2	Delays of the 24 V Outputs	4 - 6
4-3	Memory Submodule Evaluation in Status Word	4 - 11
4-4	Input Parameters of the IP 265 User Program	4 - 12
4-5	Output Parameters of the IP 265 User Program	4 - 13

4 General Operation

4.1 Parallel Program Execution with IP 265

Remember

In the case of conventional programmable controllers (PLCs), the user program for the control of an overall system is processed by a CPU. The individual instructions of the CPU user program are executed **sequentially** by the CPU.

The use of an FPGA in an S5-100U module makes it possible for the first time to handle process signals **in parallel** and therefore **very fast**.

By loading memory data (contents), hardware structures similar to the hardware wiring in SIMATIC C1, C2 and C3 systems are set up on the FPGA of the IP 265. In contrast to this, the FPGA used in the IP 265 is **programmable**, i.e. the hardware connections established on the FPGA can be "deleted" and configured again as often as desired by loading memory data.

The FPGA load data necessary for a hardware connection is defined by the user in the user program (referred to below as the IP 265 user program).

The IP 265 user program consists of basic functions such as logic operations, counters, timers and comparators and the connections necessary between these language elements. Both the language elements and the connections take up memory space (resources) on the IP 265. The resources of the IP 265 available for the language elements and their connections are limited. Only small user programs can be processed in the IP 265.

Between the IP 265 user program and the CPU user program, there are **insignificant differences** with regard to the language elements and the operands.

Note

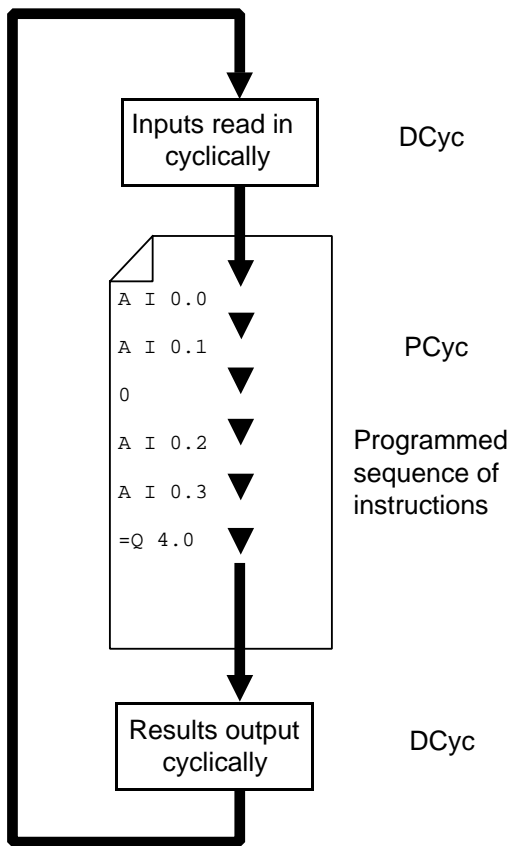
The structure of the IP 265 user program resembles the CSF 5 method of representation (programming of SIMATIC CPUs) of the CPU user program. The use of FPGA systems causes slight deviations when programming the IP 265.

There are basic differences between the parallel program execution of the IP 265 user program and the sequential program execution of the CPU user program.

Figure 4-1 shows the essential differences between sequential and parallel user program execution.

Sequential program processing

CPU user program:



Parallel program processing

IP 265 user program:

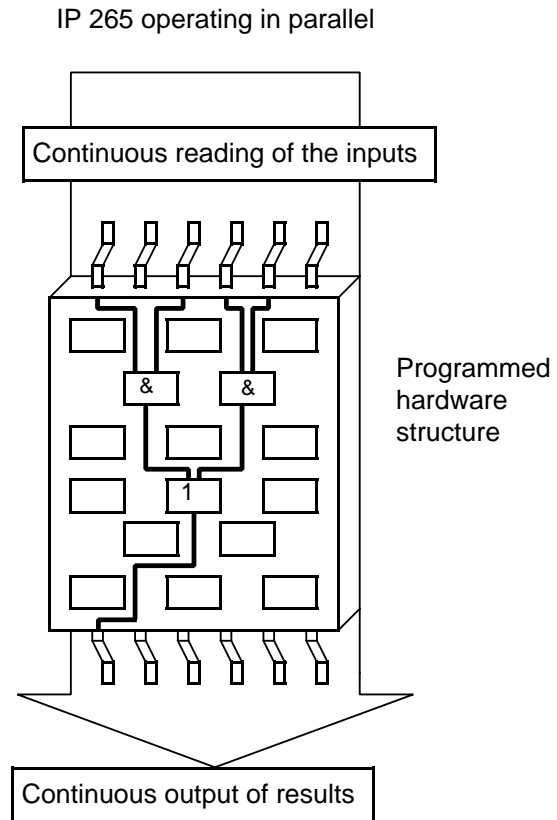


Figure 4-1. Comparison of Sequential Program Execution and Parallel Program Execution

The IP 265 can read process signals continuously. All input signals are processed by the IP 265 user program in **parallel with** output signals.

Structure of the IP 265 user program

The IP 265 user program has no block structure. The IP 265 user program is only divided into segments. In every segment, several inputs can be connected to outputs. Segments are editing aids. They do not determine the processing sequence in the IP 265.

Time characteristics of IP 265 program execution

The individual segments of the IP 265 user program are processed in parallel in the IP 265. In the case of purely binary logic, the **program execution time** is 1/128 kHz or 7.8 μs (the interconnection is clocked with 128 kHz). Every retentive language element between an input and an output of a segment leads to an additional extension of the program execution time by 7.8 μs (Section 4.2.1).

Example: Segment consisting of three binary language elements (program execution time: 7.8 μ s)
Two AND operations are ORed. I 0.0, I 0.1, I 0.2, I 0.3, Q 0.0 are the addresses of the inputs or outputs of the IP 265.

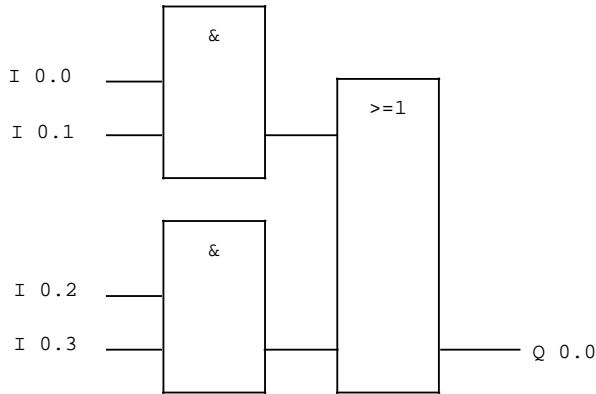


Figure 4-2. Example: Binary Segment of the IP 265 User Program

Table 4-1 gives an overview of the essential differences between CPU program execution and the IP 265 program execution.

Table 4-1. Essential Differences Between Sequential and Parallel Program Processing

Sequential program processing (CPU)	Parallel program processing (IP 265)
<ul style="list-style-type: none"> All instructions of the CPU user program are processed sequentially. 	<ul style="list-style-type: none"> The IP 265 user program is processed in parallel.
<ul style="list-style-type: none"> The CPU user program is divided into segments and blocks. The program can jump from one block to another as often as required. 	<ul style="list-style-type: none"> The IP 265 user program is only divided up into segments (for editing purposes only).
<ul style="list-style-type: none"> The speed of the CPU is basically determined by the processing rate of the instructions of the user program. The time between reading in inputs and writing of outputs (process image transfer) is the sum of the individual instruction processing times of a program run (program cycle (PCyc)). 	<ul style="list-style-type: none"> The speed of IP 265 is essentially defined by the number of retentive language elements which are connected between the inputs and outputs in the segments (timers, counters and so on).

4.2. Response Time

The response time is the time that elapses from the change of an input until setting of an output of the IP 265.

In the preceding section, we already mentioned the program execution time of the FPGA. For the assessment of the response time of the IP 265, additional delays must be considered when using the 24 V inputs and 24 V outputs.

The response time consists of the following three components:

- Input delay
- Program execution period
- Output delay

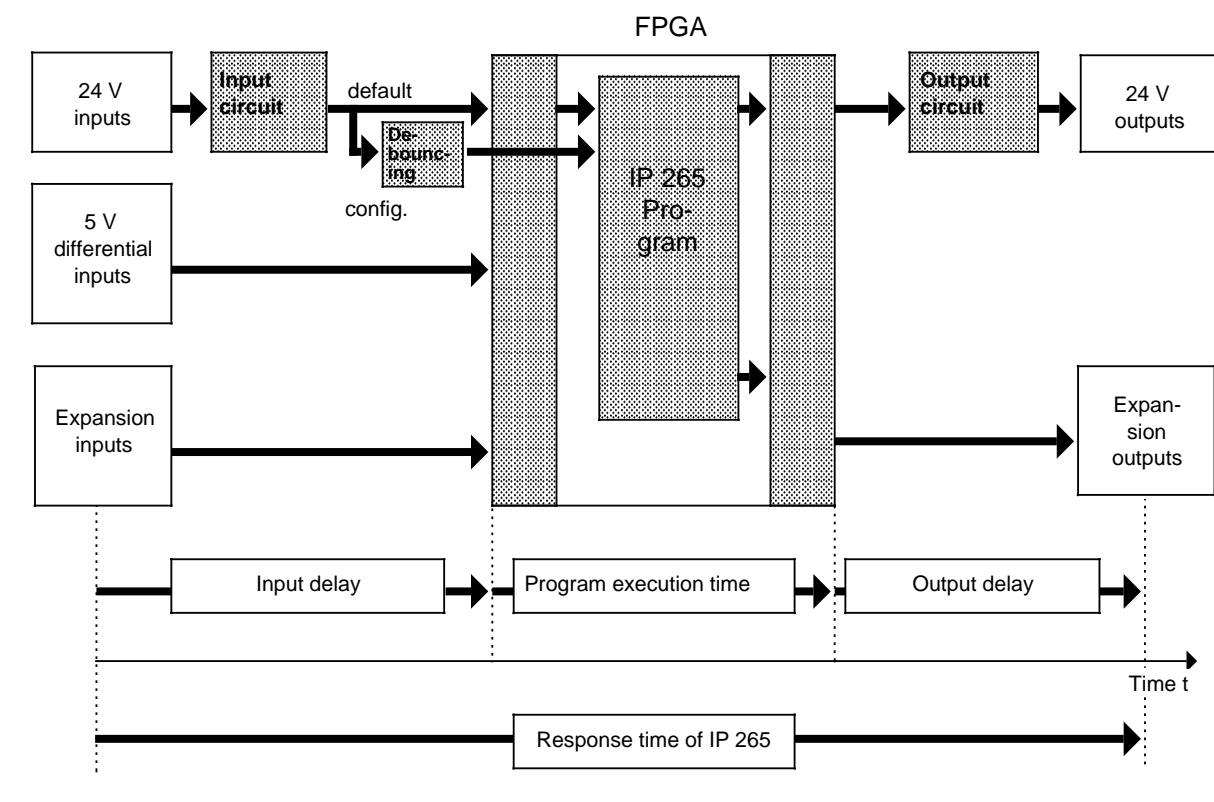


Figure 4-3. Time Diagram - Response Time of IP 265

4.2.1 Program Execution Time

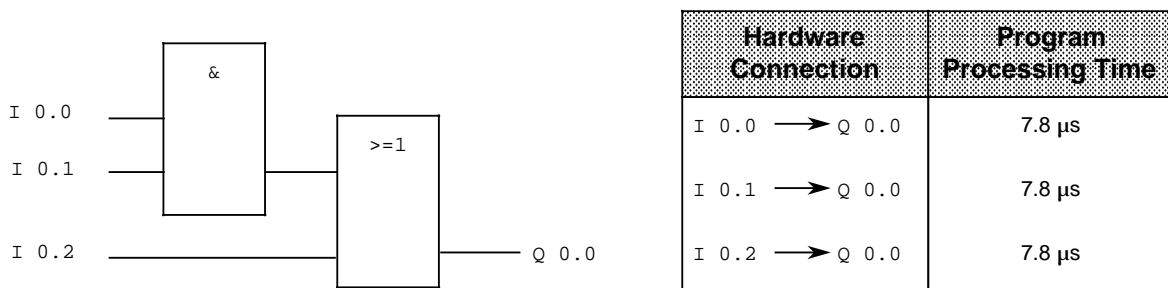
All segments of the IP 265 user program are processed in parallel in the IP 265. For purely binary operations, the program execution time is $1/128 \text{ kHz} = 7.8 \mu\text{s}$ (the interconnection is clocked with 128 kHz). Every retentive language element between an input and an output of a segment leads to an additional extension of the program execution period of $7.8 \mu\text{s}$ in each case.

Examples of retentive language elements of the IP 265:

- Timer operations
- Counter operations
- Retentive functions, such as:
 - RS flipflops
 - Binary scalers
 - Edge flags
 - Clock delay

The different program execution times in parallel paths can lead to undesired runtime effects. For synchronization of signal charts, the language element "clock delay" is available (Section 9.3.2 "Language Elements in a Control System Flowchart").

Example: Program execution times of a purely binary segment



Example: Program execution times of a segment with latching language element

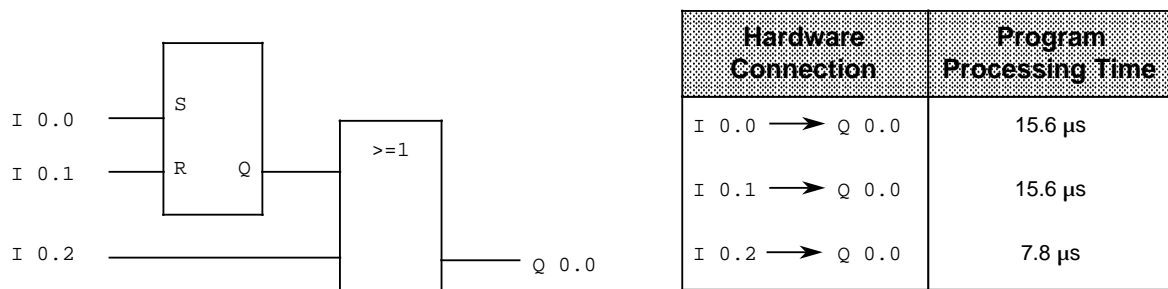


Figure 4-4. Examples: Program Execution Times in Parallel Paths of a Segment

4.2.2 Input delay

The input delay of the 24 V inputs consists of two times:

- Delay of the input circuit: typ. 15 μ s
- Debouncing time: 1 to 2 ms

Remember:

For adaptation to simple sensors, the fast 24 V inputs can be debounced. You can configure the debouncing time with the aid of COM 265.

4.2.3 Output delay

The output delay of the 24 V outputs is determined by the output wiring.

- Delay of the output wiring: typ. 70 μ s (500 mA ohmic load)

The following table gives an overview of possible output delays depending on the ohmic load of the 24 V output wiring.

Table 4-2. Delays of the 24 V Outputs

Ohmic load	Delays		Output Frequency
	Rising Edge	Falling Edge	
15 mA*	typ. 10 μ s	typ. 150 μ s	max. 1 kHz
50 mA*	typ. 10 μ s	typ. 90 μ s	max. 2 kHz
500 mA*	typ. 10 μ s	typ. 70 μ s	max. 4 kHz

* Peak value (not an r.m.s specification)

4.3 Output and Input Data

IP 265 program processing is controlled by the CPU via the external I/O bus.

For each slot, 8 bytes are reserved for the IP 265 in the process image of the outputs (PIQ) and in the process image of the inputs (PII) of the CPU. 8 bytes of output data can be stored for the IP 265 in the PIQ. The IP 265 on the other hand provides 8 bytes of input data for the CPU for further evaluation in the PII.

The PIQ can be assigned the following output data:

- Word 0: **Control word:** Instructions for IP 265, e.g. "Switch IP 265 to RUN"
- Word 2, 4, 6: **Input parameters** of the IP 265 user program

The following IP 265 input data can be evaluated in the PII:

- Word 0: **Status word:** Status of the IP 265 or of program scanning, e.g. "IP 265 in RUN"
- Word 2, 4, 6: **Output parameter** of the IP 265 user program

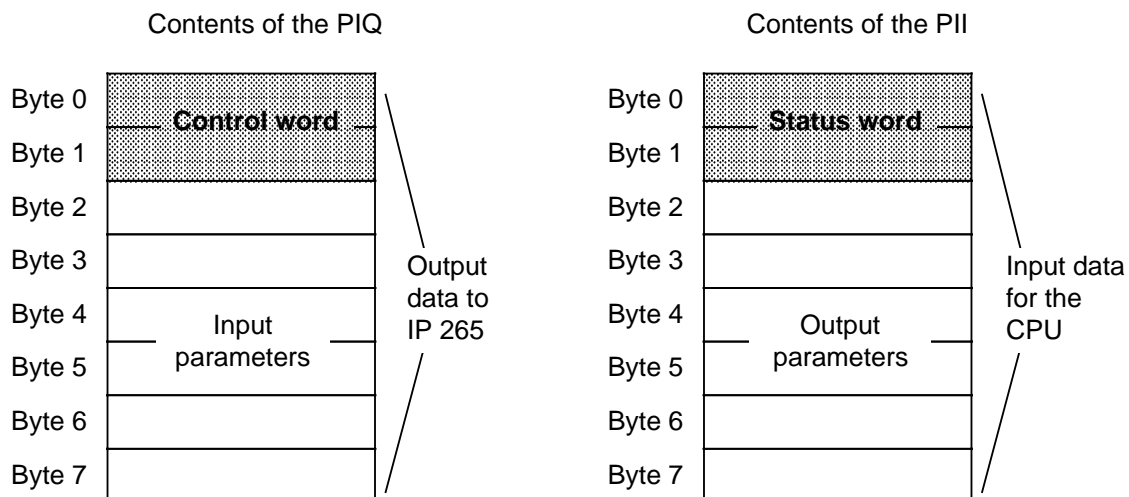


Figure 4-5. Assignment of the PIQ and PII in the CPU

The PIQ and the PII can be written and read

- by the CPU user program (normal operation)
- with the aid of COM 265 (on-line test)
- with the aid of the "FORCE VAR" PLC function

4.3.1 Structure and Operation of the Control Word

Control commands are transferred to the IP 265 via the control word. The IP 265 can thus be switched to the STOP and RUN operating states.

Two types of instruction can be stored in the control word:

- Instructions reserved for COM 265
- Instructions accessible to the user (CPU user program).

Note

When the IP 265 is being checked by COM 265 (COM 265 instruction active), the CPU user program may not write the output data for the IP 265 in the PIQ. This must be taken into consideration for program generation (scan of the control bit in the status word of the IP 265; Section 4.3.2).

General structure of the control word:

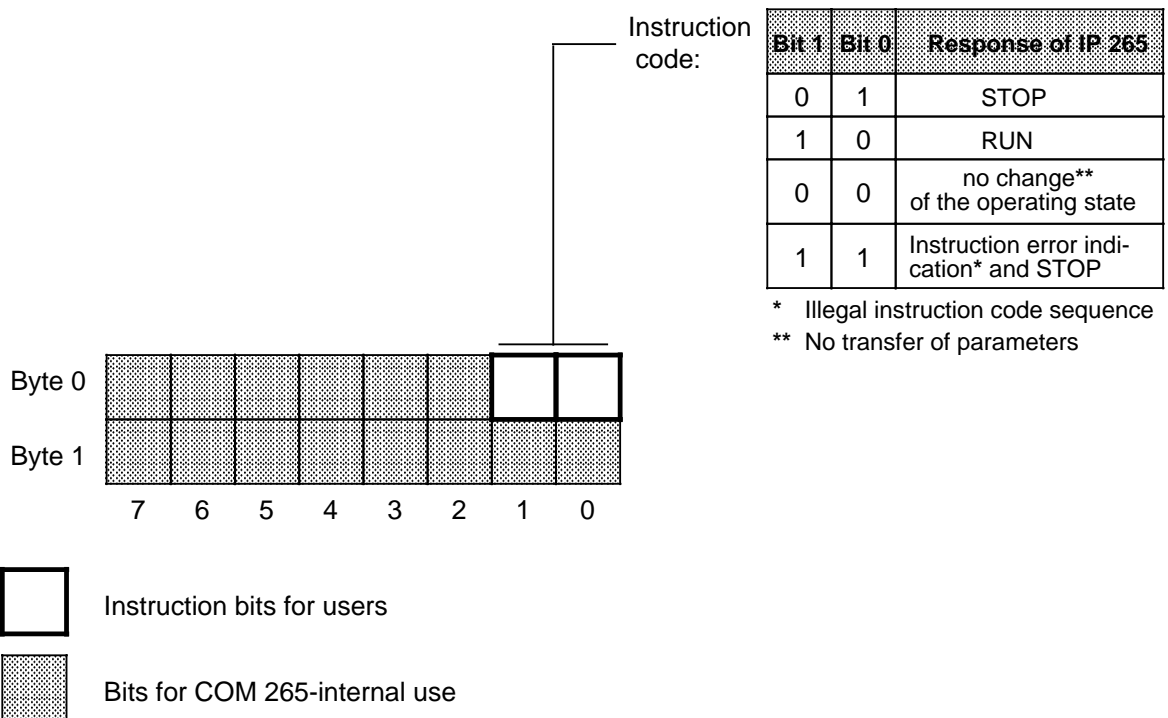


Figure 4-6. Structure of the Control Word

Note

Only the 2 instruction bits in byte 0 of the control word are relevant for the control of the IP 265 by the CPU user program. Only one of the two bits may be set, where bit 0 is the STOP bit and bit 1 is the RUN bit. The remaining bits of the control word are handled independently by COM 265 and must **not be** overwritten by the user.

4.3.2 Structure and Handling of the Status Word

The operating state and possible error states of the IP 265 can be recognized via the status word.

You can evaluate the following in the status word of the IP 265

- The status of IP 265,
- Possible hardware errors of the IP 265,
- Loading errors of the IP 265 and
- Illegal instructions in the control word.

General structure of the status word:

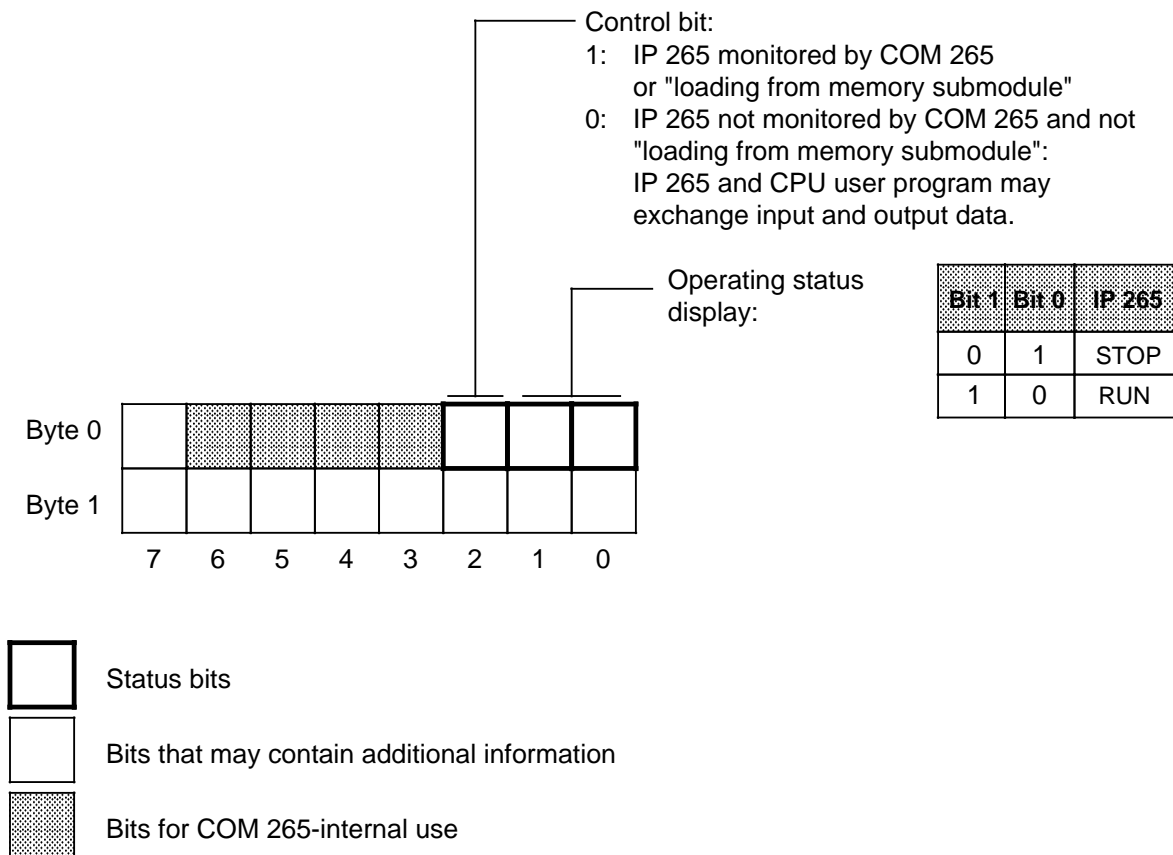


Figure 4-7. Structure of the Status Word

Note

The information in the status word is always a response to the control data of the preceding data cycle. However, they are not an "echo" of the instruction in the control word. Bits 0 and 1 in byte 0 of the status word always display the current operating state of the IP 265.

The control bit (bit 2 in byte 0 of the status word) is set

- after POWER ON while the IP 265 is being loaded from the memory submodule,
- during loading of the IP 265 via the remote I/O bus and
- shortly after selection of the COM 265 function "on-line test".

Status and error indication

In the case of an error, the IP 265 goes into STOP (exception: short-circuit at 24 V output). In addition to the display of the operating state, the IP 265 provides error messages and further status information.

Further status and error indications in the status word:

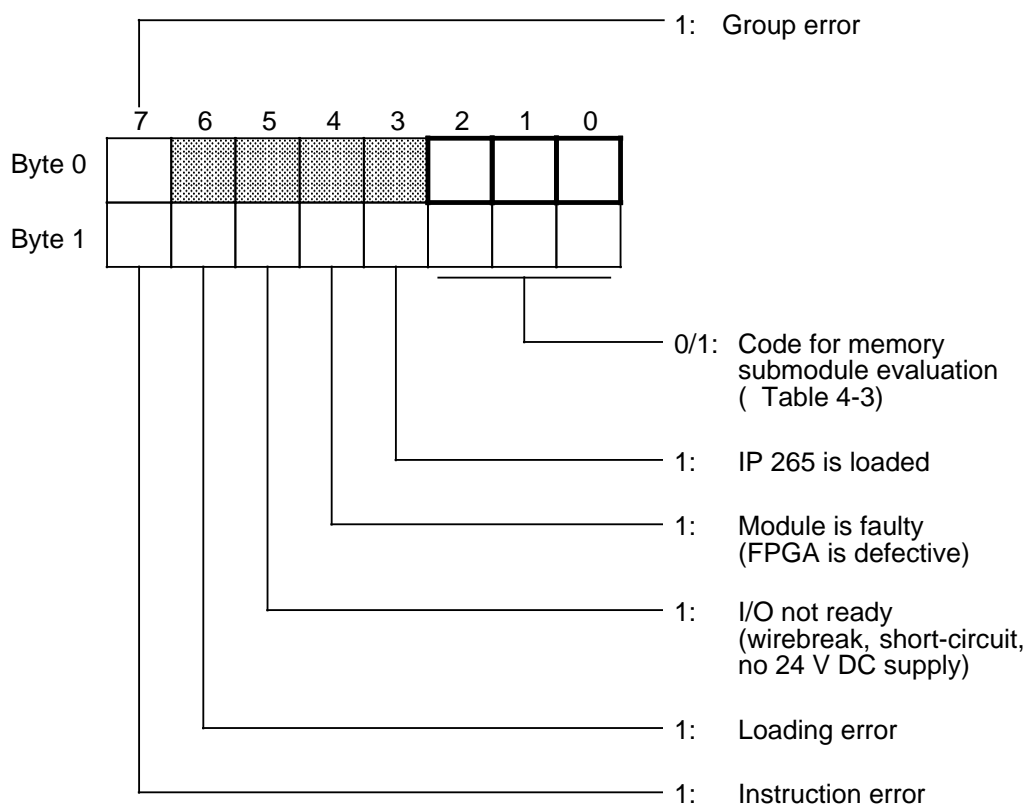


Figure 4-8. Error Messages and Status Information in the Status Word

Group error: The group error bit is set for every error.

Instruction error: If the instruction error bit is set, the instruction in the control word of the preceding data cycle has not been processed. It shows the following:

- A useless instruction; e.g. RUN and STOP bits are set concurrently in the control word (signal state "1")
- Incorrect parameter values (time values, counts outside the permissible range of values: 0 to 999)

Memory submodule evaluation: After POWER ON, the IP 265 executes a memory submodule evaluation during restart. The relevant information is displayed in bits 0 to 2 in byte 1 of the status word with a corresponding code (Table 4-3).

Table 4-3. Memory Submodule Evaluation in Status Word

Code in Byte 1 of the Status Word (Bit 0 to 7)			Message of Memory Submodule Evaluation	Operating State of IP 265
7	6	5 4 3 2 1 0		
	0	0 0	No memory submodule connected	STOP
	0	0 1	Wrong memory submodule connected (not an S5-90 submodule)	Hard STOP
	0	1 0	Wrongly programmed or empty EPROM memory submodule connected	Hard STOP
	0	1 1	EPROM memory submodule with valid IP 265 program	STOP
	1	0 0	Empty EEPROM memory submodule connected	STOP
	1	0 1	Wrongly programmed EEPROM memory submodule connected	STOP
	1	1 0	EEPROM memory submodule with valid IP 265 program connected	STOP
	1	1 1	Memory submodule inserted or removed during POWER ON.	Hard STOP

Note

The IP 265 can only exit a hard STOP by switching to POWER OFF.

4.3.3 Parameters of the IP 265 User Program

Parameters can be exchanged between CPU user program and IP 265 user program via the PIQ and the PII:

- The CPU user program can write input parameters (6 bytes) into the PIQ where they are read by the IP 265.
- The IP 265 can provide IP 265 program results in the form of output parameters (6 bytes) which are read by the CPU user program in the PII and evaluated.

For IP 265 and CPU program generation, the parameters must be defined accordingly and supplied with parameter data. In Section 13 "Sample Applications and Programming Aids", you will find examples of this.

Input parameters

Input parameters guarantee the supply of the IP 265 user program by the CPU user program.

Note

The input parameters are only processed by the IP 265 if the RUN bit is set in the control word.

The following input parameters can appear in the input bar of the control system flowchart (IP 265 user program):

Table 4-4. Input Parameters of the IP 265 User Program

Possible Input Parameters	Data Type	Permissible Range
Count, time value	10 bit count	KF +(0 to 999)
	10 bit time value	KF +(0 to 999)
Bit input parameter	Bit	Signal "0" or "1"

The counters and timers in the IP 265 can be supplied with constants by the CPU user program. However, it is not possible to work with the SIMATIC S5 constants for timers (KT) and counters (KC). Constant count and time values can only be transmitted in BCD code from the CPU to the IP 265.

Output parameters

Output parameter are results of IP 265 program scanning which are read and processed by the CPU user program.

Note

The output parameters of the IP 265 user program at the time of an IP 265 STOP can be read and evaluated at any time by the CPU user program.

The following output parameters can appear in the output bar of the control system flowchart (IP 265 user program):

Table 4-5. Output Parameters of the IP 265 User Program

Possible Output Parameters	Data Type	Permissible Range
Count	10 bit count	KF +(0 to 999)
Bit output parameter	Bit	Signal "0" or "1"

5 Addressing		
5.1	Addressing in the IP 265 User Program (IP 265 Viewpoint)	5 - 2
5.1.1	Actual Addresses of the Inputs and Outputs of the IP 265	5 - 2
5.1.2	Actual Addresses of the Input and Output Parameters (IP 265 Viewpoint)	5 - 3
5.2	Addressing in the CPU User Program (CPU Viewpoint)	5 - 4
5.2.1	Addressing of the Control Word and the Status Word	5 - 5
5.2.2	Actual Addresses of the Input and Output Parameters (CPU Viewpoint)	5 - 6
5.3	Allocation of the Parameter Addresses in the IP 265 User Program and in the CPU User Program	5 - 8

Figures		
5-1	Communications Model	5 - 1
5-2	Example: Addressing the Status Word in the CPU User Program	5 - 5
5-3	Example: Addressing an Input Parameter in the CPU User Program	5 - 7
Tables		
5-1	Addresses of the Inputs and Outputs of the IP 265	5 - 2
5-2	Addresses of the Input Parameters in the IP 265 User Program	5 - 3
5-3	Addresses of the Output Parameters in the IP 265 User Program	5 - 3
5-4	Byte Addresses in the PII/PIQ (CPU Viewpoint)	5 - 4
5-5	Addresses of the Input Parameters in the CPU User Program	5 - 6
5-5	Addresses of the Output Parameters in the CPU User Program	5 - 6
5-6	Addresses of the Input Parameters of the IP 265	5 - 8
5-7	Addresses of the Output Parameters of the IP 265	5 - 8

5 Addressing

Remember

The IP 265 is provided with

- interfaces to the process,
- a port to a further IP 265 and
- a port to the I/O bus for communication with the CPU.

Figure 5-1 shows the relation between the hardware ports of the IP 265 and the margin bar objects of the control system flowchart (IP 265 user program). Possible operands of the control system flowchart are in the margin bars.

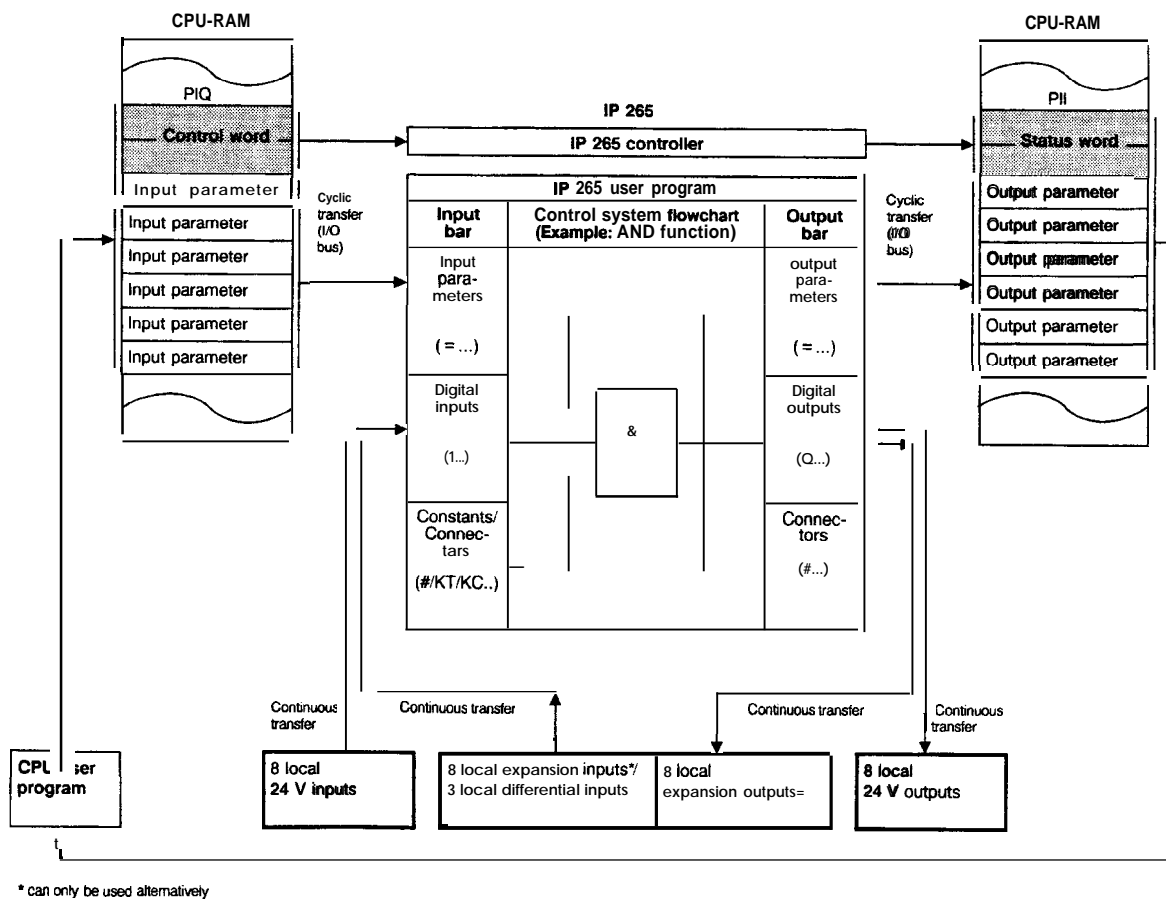


Figure 5-1. Communications Model

5.1 Addressing in the IP 265 User Program (IP 265 Viewpoint)

All interfaces of the IP 265 are addressed in the IP 265 user program with fixed actual addresses:

- 24 V inputs and 24 V outputs
- 5 V differential inputs
- Expansion inputs/outputs
- Parameter interface

5.1.1 Actual Addresses of the Inputs and Outputs of the IP 265

Note

The actual addresses of the inputs and outputs of the IP 265 are slot-independent in the IP 265 user program.

Table 5-1. Addresses of the Inputs and Outputs of the IP 265

Inputs/Outputs	Addresses
24 V inputs	I 0.y; y (0 to 7)
24 V outputs	Q 0.y; y (0 to 7)
Differential inputs*	I 1.y; y (0 to 2)
Expansion inputs*	I 2.y; y (0 to 7)
Expansion outputs*	Q 2.y; y (0 to 7)

* can only be used alternatively

The differential inputs and the expansion inputs/outputs are connected to the same 15-pole sub D socket. The differential inputs and the expansion interface cannot be used simultaneously.

You may use each of the 8 pins of the expansion interface either as expansion input or as expansion output (configurable with COM 265).

Note

The inputs and outputs of the IP 265 are local and therefore not directly accessible via the normal address space of the CPU. Their allocated fixed addresses do not appear in the CPU user program.

5.1.2 Actual Addresses of the Input and Output Parameters (IP 265 Viewpoint)

Note

The actual addresses of the input and output parameters of the IP 265 are slot-independent in the IP 265 user program.

The addresses (direct addressing) are used for addressing the parameter area of a maximum of 6 bytes for input parameters and 6 bytes for output parameters.

**Table 5-2. Addresses of the Input Parameters
in the IP 265 User Program**

Possible Input Parameters	Data Type	Direct Addressing in the IP 265 User Program (Slot-Independent)
Count, time value	10 bit count, 10 bit time value	IW x. ; x (66, 68, 70)
Bit input parameter	Bit	I x.y ; x (66 to 71) y (0 to 7)

**Table 5-3. Addresses of the Output Parameters
in the IP 265 User Program**

Possible Output Parameters	Data Type	Direct Addressing in the IP 265 User Program (Slot-Independent)
Count	10 bit count	QW x. ; x (66, 68, 70)
Bit output parameter	Bit	Q x.y ; x (66 to 71) y (0 to 7)

In Section 5.3, you will find the assignment of the parameter addresses in the IP 265 user program to the parameter addresses in the CPU user program.

5.2 Addressing in the CPU User Program (CPU Viewpoint)

The CPU accesses the IP 265 via the process image of the inputs (PII) and the process image of the outputs (PIQ). The CPU addresses the IP 265 in the address space of the analog channels. The address space is determined by the slot selected for the IP 265.

The address area of the module comprises 8 bytes of input data and 8 bytes of output data. Input and output data are addressed via the same address space.

The reserved 8 bytes have a fixed meaning:

Output data for IP 265 is stored by the CPU user program in the process image of the outputs (PIQ):

- Control word (2 bytes)
- Input parameters of the IP 265 user program (6 bytes)

In the process image of the inputs (PII), input data is provided by the IP 265 for evaluation in the CPU user program:

- Status word (2 bytes)
- Output parameters of the IP 265 user program (6 bytes)

Note

The control word, status word and parameters of the CPU can be addressed bit by bit, byte by byte or word by word.

Please consider the following for addressing:

- The module may only be used in slots 0 to 7.
- The address space ranges from byte 64 to byte 127.
- For each slot, 8 bytes are reserved both in the process image of the inputs (PII) and in the process image of the outputs (PIQ) of the CPU.

Table 5-4. Byte Addresses in the PII/PIQ (CPU Viewpoint)

Slot	0	1	2	3	4	5	6	7
Addresses PII/PIQ	64 to 71	72 to 79	80 to 87	88 to 95	96 to 103	104 to 111	112 to 119	120 to 127
<input type="checkbox"/> Starting address of a slot								

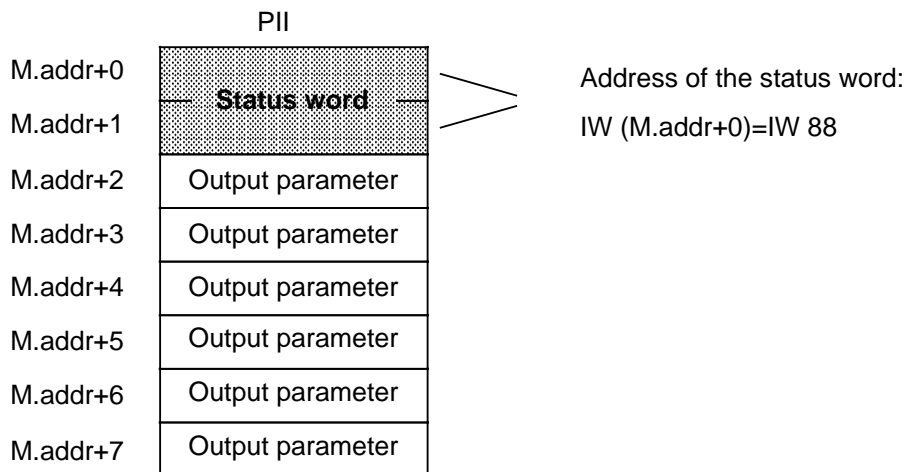
5.2.1 Address of the Control Word and the Status Word

Note

The addresses for the control word and status word of the IP 265 are slot-dependent in the CPU user program.

The address of the status word or control word always corresponds to the starting address of the module slot. For the starting addresses of the possible slots for the IP 265, see Table 5-4.

Example: Determine actual address of the status word
 You want to evaluate the status word in the CPU user program. You have plugged the IP 265 into slot 3. Consequently, the starting address of the slot (module address) is 88.



M.addr=Module address

Figure 5-2. Example: Addressing the Status Word in the CPU User Program

5.2.2 Actual Addresses of the Input and Output Parameters (CPU Viewpoint)

Note

The addresses of the input and output parameters of the IP 265 are slot- dependent in the CPU user program. For address allocation, you must add the starting address of the slot into which you have plugged the module (module address) to the relevant byte number in the address area (byte 2 to 7).

The addresses (direct addressing) are used to address the parameter area of a maximum of 6 bytes for input parameters and 6 bytes for output parameters.

Table 5-5. Addresses of the Input Parameters in the CPU User Program

Possible Input Parameter	Data Type	Direct Addressing in the CPU User Program (Slot-Dependent)
Count, time value	10 bit count, 10 bit time value	QW z ; z (M.addr*+2, M.addr+4, M.addr+6)
Bit input parameter	Bit	Q z.y ; z (M.addr*+2 to M.addr+7) y (0 to 7)

* M.addr=Module address (Table 5-4)

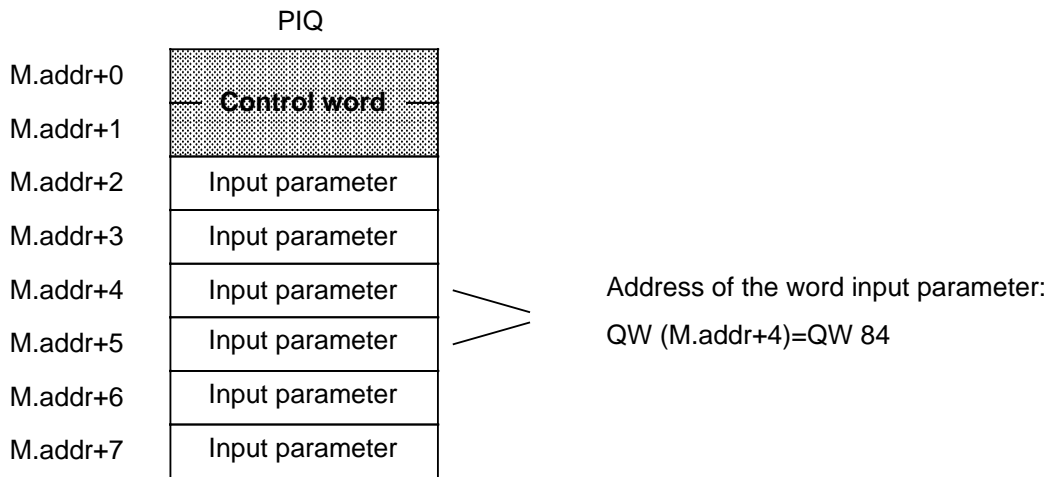
Table 5-6. Addresses of the Output Parameters in the CPU User Program

Possible Output Parameters	Data Type	Direct Addressing in CPU User Program (Slot-Dependent)
Count	10 bit count	IW z ; z (M.addr*+2, M.addr+4, M.addr+6)
Bit output parameter	Bit	I z.y ; z (M.addr*+2 to M.addr+7) y (0 to 7)

* M.addr=Module address (Table 5-4)

Section 5.3 includes an allocation of the parameter addresses in the CPU user program to the parameter addresses in the IP 265 user program.

Example: Determining the address of an input parameter
You want to store e.g. a count in bytes 4 and 5 of the PIQ in the CPU user program.
You have plugged the IP 265 into slot 2. Consequently, the starting address of the slot (module address) is 80.



M.addr=Module address

Figure 5-3. Example: Addressing an Input Parameter in the CPU User Program

5.3. Allocation of the Parameter Addresses in the IP 265 User Program and in the CPU User Program

In order to prevent programming errors, the tables below give you an overview of the correct addressing of the parameters of the IP 265 (repetition from the preceding sections).

The information given in Tables 5-7 and 5-8

- illustrates the differences in programming of the parameters in the IP 265 user program (IP 265 viewpoint) and in the CPU user program (CPU viewpoint),
- allocates the parameter addresses in the CPU user program to the IP 265 user program

Table 5-7. Addresses of the Input Parameters of the IP 265

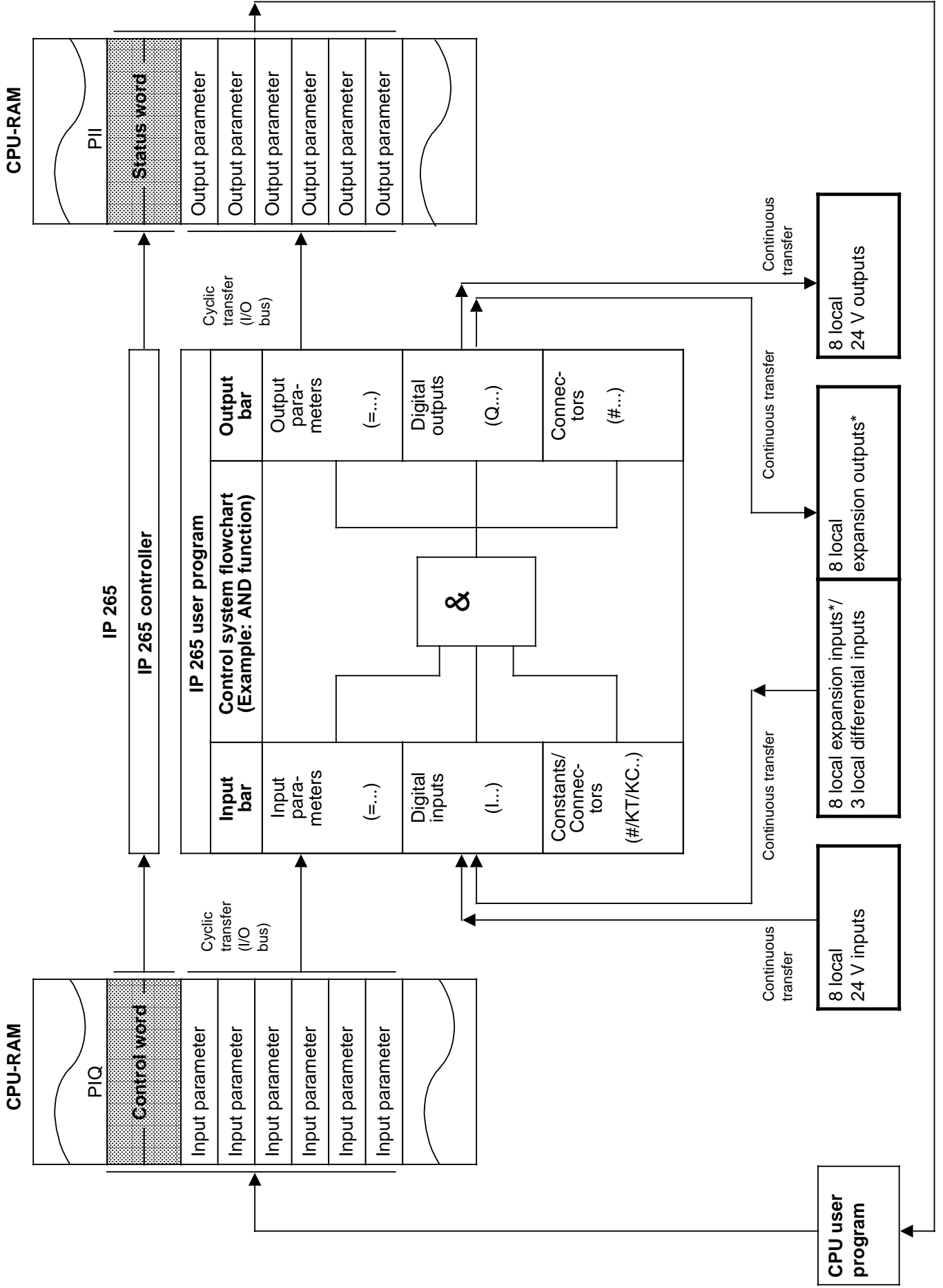
Possible Input Parameters	Data Type	Direct Addressing in the CPU User Program (Slot-Dependent)	Direct Addressing in the IP 265 User Program (Slot-Independent)
Count, time value	10 bit count, 10 bit time value	QW z ; z (M.addr *+2, M.addr+4, M.addr+6)	IW x ; x (66, 68, 70)
Bit input parameter	Bit	Q z.y ; z (M.addr *+2 to M.addr+7) y (0 to 7)	I x.y ; x (66 to 71) y (0 to 7)

* M.addr=Module address (Table 5-4)

Table 5-8. Addresses of the Output Parameters of the IP 265

Possible Output Parameters	Data Type	Direct Addressing in the CPU User Program (Slot-Dependent)	Direct Addressing in the IP 265 User Program (Slot-Independent)
Count	10 bit count	IW z ; z (M.addr *+2, M.addr+4, M.addr+6)	QW x ; x (66, 68, 70)
Bit output parameter	Bit	I z.y ; x (M.addr *+2 to M.addr+7) y (0 to 7)	Q x.y ; x (66 to 71) y (0 to 7)

* M.addr=Module address (Table 5-4)



* can only be used alternatively

6	Control of the IP 265 by the CPU User Program	
6.1	Control of IP 265 Program Execution	6 - 1
6.2	Transfer Times for Data Exchange Between CPU and IP 265	6 - 3
6.3	CPU Program Section for Control of the IP 265	6 - 4
6.3.1	Structure of the CPU Program Section	6 - 5
6.3.2	Programming blocks of the CPU Program Part	6 - 6

Figures		
6-1	Control System Flowchart: Control of IP 265 by the CPU User Program . . .	6 - 2
6-2	Transfer of Input and Output Data	6 - 3
6-3	Structure of the CPU Program Section for Control of the IP 265	6 - 5
Tables		
6-1	Transfer Times Between IP 265 and CPU	6 - 4

6 Control of the IP 265 by the CPU User Program

6.1 Control of IP 265 Program Execution

Program execution in the IP 265 is controlled by the CPU user program. The IP 265-related part of the CPU user program communicates with the IP 265 via the PIQ and PII of the CPU:

- In the process image of the outputs (PIQ), the CPU user program stores commands (control word) intended for the controller of the IP 265.
- In the process image of the inputs (PII), the CPU user program reads out the status of the IP 265 (status word).
- In normal operation of the IP 265, parameters can be exchanged between the CPU user program and the IP 265 user program via the PII and the PIQ for the following purposes:
 - Transfer of the results of logic operations from the CPU to the IP 265 and vice versa (results of logical operations in the user program)
 - Loading time values and counts of the CPU into the IP 265
 - Reading counts from the IP 265 into the CPU

Note

The transfer of the parameters requires additional program capacity of the IP 265.

The CPU user program may only write input parameters into the PIQ if the IP 265 is not controlled by COM 265 and is therefore "enabled" for communication with the CPU. This state is indicated by the control bit in the status word (Section 4.3.2).

To ensure correct IP 265 program scanning, the following note is of particular importance for control of the IP 265 by the CPU user program:

Note

Before every transfer of new input parameters to the IP 265, the status word of the IP 265 must be evaluated in the CPU user program.

The representation in Figure 6-1 shows the control of the IP 265 by the CPU user program. Normal operation is represented, i.e. a memory submodule with tested IP 265 user program is plugged into the IP 265.

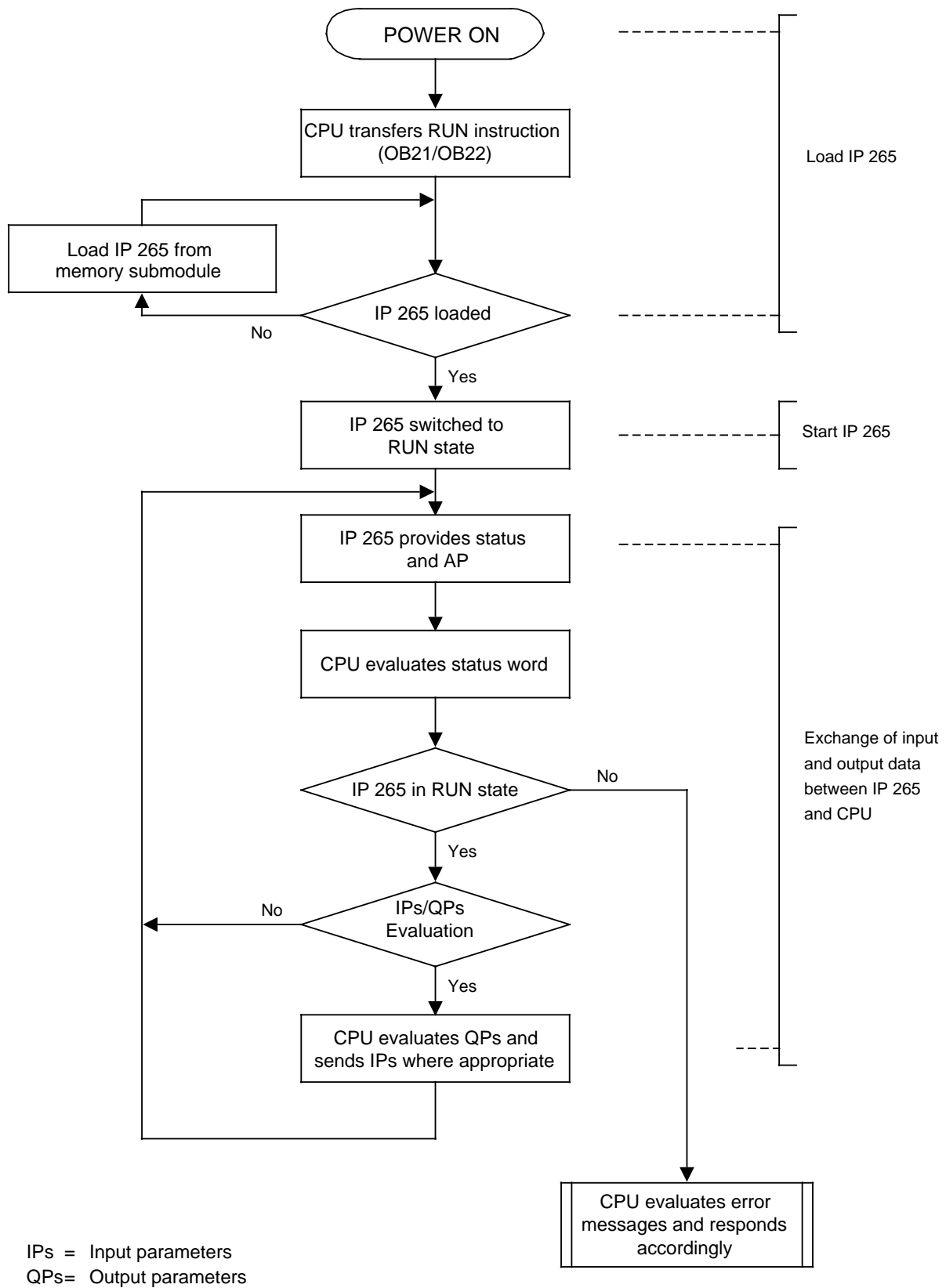


Figure 6-1. Control System Flowchart: Control of IP 265 by the CPU User Program

6.2 Transfer Times for Data Exchange Between CPU and IP 265

Input and output data is transferred **continuously** between the IP 265 and the I/O bus. If the IP 265 is in the RUN state,

- the output data of the I/O bus (8 bytes) is read into the IP 265,
- the IP 265 checkback signals are written into the input data area (8 bytes) over the I/O bus.

Data exchange via the I/O bus **is carried out cyclically** by the CPU. The input and output data is transferred between the IP 265 and the CPU in 8-byte data blocks and in one data cycle (OB1: S5-100 Manual) .

Remember:

A CPU scan cycle consists of two separate processes:

- **Program cycle (PCyc)**
The STEP 5 statements of the CPU user program are executed.
The I/O bus is inactive.
- **Data cycle (DCyc)**
The input and output data is transferred via the I/O bus between the CPU and the IP 265.

Program cycle and data cycle constantly alternate.

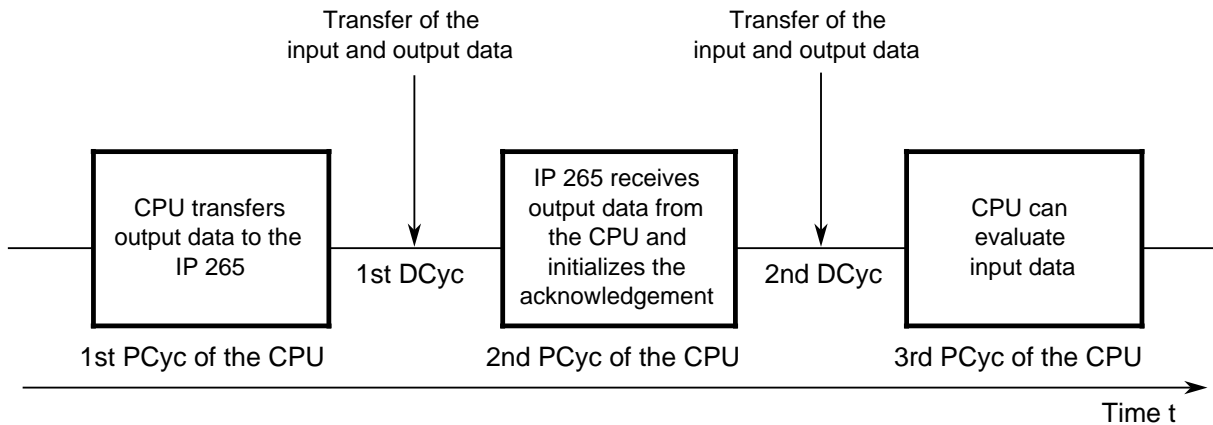


Figure 6-2. Transfer of Input and Output Data

Note

The scan cycle diagram shows that a response of the IP 265 to changes in the output data (control word, input parameter) are evaluated by the CPU user program in the next-but-one program cycle at the earliest. This can have consequences for programming the user program.

Summary

The transfer times for the data exchange between the CPU and the IP 265 (t_{Tr}) are basically determined by the program scan time (t_{PCyc}) and the data transfer time (t_{DCyc}) of the CPU (system characteristic of the S5-100 system family).

Table 6-1. Transfer Times Between IP 265 and CPU

Data Transfer	Transfer Time
IP 265 user program CPU user program	$t_{Tr} \cdot 2 \cdot t_{PCyc} + 2 \cdot t_{DCyc}$
CPU user program IP 265 user program	$t_{Tr} \cdot t_{PCyc} + t_{DCyc}$

t_{PCyc} =Program cycle time, t_{DCyc} =data cycle time

6.3 CPU Program Section for Control of the IP 265

A program section is listed in STL on the following page which gives an **example** of the control of the IP 265 by the CPU user program. In Figure 6-1, this program section represents the conversion of the function model "Control of the IP 265 by the CPU user program".

The CPU program includes

- switching the IP 265 to the RUN state,
- exclusion of COM 265 control of the IP 265 and
- parameter entry/deletion in the parameter area (6 bytes).

Error evaluations are not dealt with in the following CPU program section.

The CPU user program controls the IP 265 for the most part via individual bits which change the operating state or which indicate the status of the module. The parameters are exchanged by means of load and transfer operations via flag areas.

Note

Before you copy-type the following programming blocks, check

- whether the slot of your module matches the slot in the example;
if not, change to another module slot or adapt the addresses in the blocks to the address area of your IP 265;
- whether you want to exchange parameters between the IP 265 and the CPU;
if not, you need not retype FB1 "CONTROL", FB2 "IPPARAM";
- whether you want to enter/delete only output parameters or only input parameters;
if yes, you only need to retype the relevant program half of the FB2 "IPPARAM".

6.3.1 Structure of the CPU Program Section

The CPU program section for controlling the IP 265 consists of 3 nesting levels. The function blocks called by the OB1 are processed cyclically.

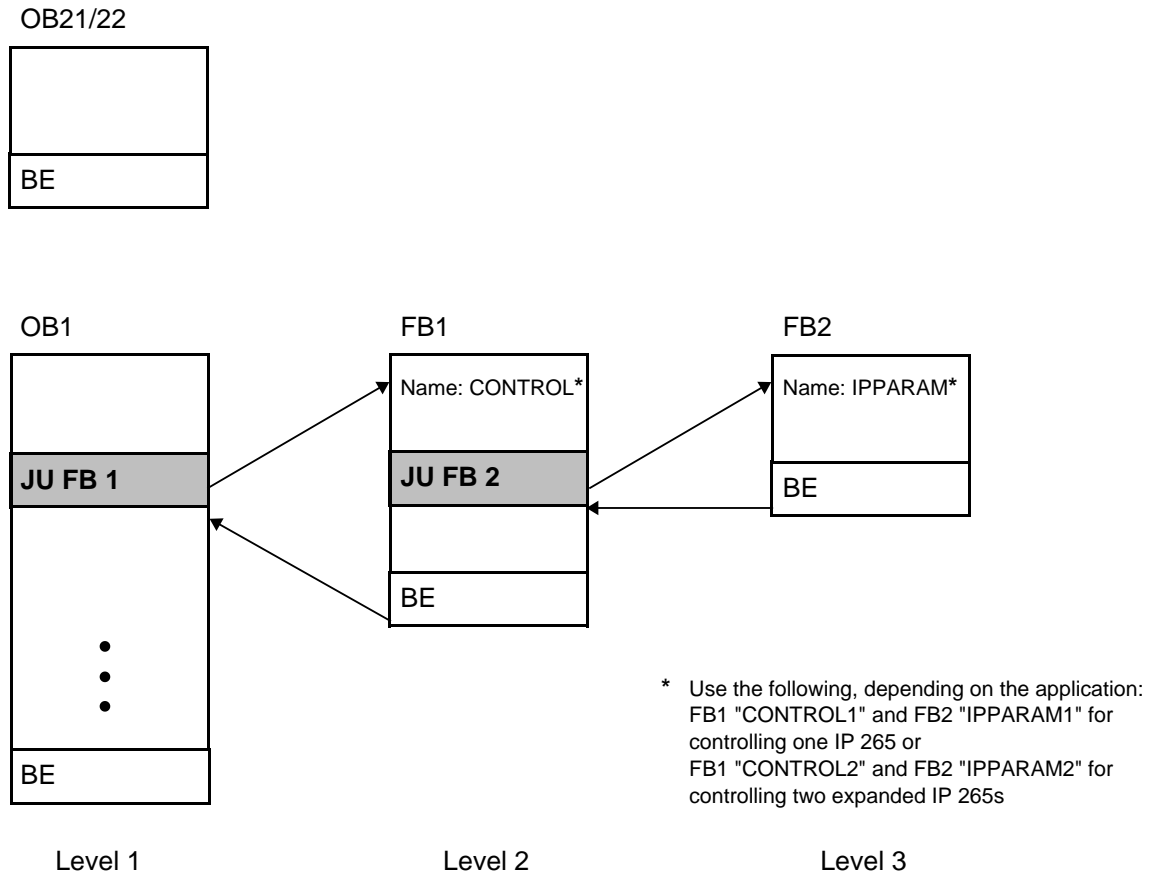


Figure 6-3. Structure of the CPU Program Section for Control of the IP 265

- **OB21/22:**
The RUN command for the IP 265 is given during restart of the CPU.
- **FB1 "CONTROL":**
The FB1 "CONTROL" checks the operating state of the IP 265. If the IP 265 is not controlled by COM 265 and switched to RUN, the FB2 "IPPARAM" is processed. For an expansion, the operating states of both IP 265 modules must additionally be synchronized (operating state adjustment: Section 12).
- **FB2 "IPPARAM":**
The FB2 "IPPARAM" supports parameter exchange between CPU and IP 265. Parameter exchange via flag areas is initiated only if the IP 265 is ready. The following happens at one position in the CPU user program during a CPU program cycle:
 - Output parameters (6 bytes) of the IP 265 are written from the PI1 into a flag area of the CPU RAM and
 - Input parameters (6 bytes) of the IP 265 are written from the flag area into the PIQ.

Advantage of parameter exchange via flag areas:

The CPU can access the parameters at any location in the CPU user program without having to check each time whether the IP 265 is ready.

6.3.2 Programming Blocks of the CPU Program Part

We provide you with the following function blocks:

- OB21/22: Switch the IP 265 to the RUN state at restart
- OB1: Invoke the control FB (FB1 "CONTROL1" or FB1 "CONTROL2")
- FB1 "CONTROL1": STL for control of **one** IP 265 (slot 2),
- FB1 "CONTROL2": STL for control of **two** IP 265 modules which have ben expanded (slots 2 and 3)
- FB2 "IPPARAM1": STL for parameter exchange with **one** IP 265 (slot 2)
- FB2 "IPPARAM2": STL for parameter exchange between **two** IP 265 modules which are expanded (slots 2 and 3)

Control of an IP 265

OB 21/22

```

Segment 1      0000
0000      :L   KH  0200      Set RUN bit of the IP 265
0001      :T   QW  80      during restart of the CPU
0002      :
.
.
.
ABCD      :BE

```

Normal CPU user program:
without processing input
and output data of IP 265

OB 1

```

Segment 1      0000
0000      :JU FB 1          Call of FB 1:
0001 NAME    :CONTROL1     Evaluate status word
0002      :
.
.
.
ABCD      :BE

```

Normal CPU user program:
without processing of input
and output data of IP 265

FB 1

```

Segment 1      0000      IP 265 in slot 2
Name :CONTROL1

000A      :A   I   80.1      IP 265 in RUN state and
000B      :AN  I   80.2      IP 265 not controlled by COM 265:
000C      :JC  FB  2      --> Assign and delete parameters
000D NAME    :IPPARAM1     of IP 265
000E      :BE

```

FB 2

```

Segment 1      0000      IP 265 in slot 2
Name :IPPARAM1

0008      :L   IW  82      Load output parameters of IP 265
0009      :T   FW  10      into slot 2 and store
000A      :L   IW  84      in flag words 10 to 14
000B      :T   FW  12      for processing
000C      :L   IW  86      by the CPU user program
000D      :T   FW  14
000E      :
000F      :L   FW  16      Transfer flag words 16 to 20 to
0010      :T   QW  82      the input parameters of IP 265
0012      :L   FW  18      in slot 2
0013      :T   QW  84
0014      :L   FW  20
0015      :T   QW  86
0016      :BE

```

Control of two IP 265 modules which have been expanded

OB 21/22

```

Segment 1      0000
0000      :L   KH  0000      Auxiliary flag
0001      :T   FY  64
0002      :L   KH  0200      Set RUN bit of both IP 265 modules
0003      :T   QW  80      at restart of CPU
0004      :L   KH  0200
0005      :T   QW  88
.
.
.
ABCD      :BE

```

Normal CPU user program:
without processing input
and output data of IP 265

OB 1

```

Segment 1      0000
0000      :JU FB 1      Invoke FB 1:
0001 NAME :CONTROL2      Assign control word and evaluate
0002      :      status word
.
.
.
ABCD      :BE

```

Normal CPU user program:
without processing of input
and output data of IP 265

FB 1

```

Segment 1      0000      IP 265 modules in slots 2 and 3
Name :CONTROL2

000A      :AN   F  64.3      Wait cycles elapsed?
000B      :JC   =M001

```

(FB 1, continued)

```

000C      :A   I   80.2      Is one of the two IP 265 modules
000D      :O   I   88.2      controlled by COM 265:
000E      :BEC                                     --> BEC
0010      :A   I   80.0      If one of two IP 265 modules
0011      :O   I   88.0      is in STOP state:
0012      :S   Q   80.0      --> set other IP 265 to STOP, too,
0013      :S   Q   88.0      and
0014      :R   Q   80.1      reset RUN bit in control word of
0015      :R   Q   88.1      of both IP 265 modules
0016      :BEC
0017      :JU   FB   2      --> Assign/delete parameters of IP 265
0018 NAME :IPPARAM2      in slots 2 and 3
0019      :BEU
001A M001 :L   FY   64      Wait cycles: CPU at restart
001B      :L   KF   +1
001C      :+F
001D      :T   FY   64
001E      :BE

```

FB 2

```

Segment 1      0000      IP 265 in slot 2
Name :IPPARAM2

0008      :L   IW   82      Load output parameters of IP 265
0009      :T   FW   10      in slot 2 and store in
000A      :L   IW   84      flag words 10 to 14
000B      :T   FW   12      for processing
000C      :L   IW   86      by the CPU user program
000D      :T   FW   14
000E      :
000F      :L   FW   16      Transfer flag words 16 to 20
0010      :T   QW   82      to input parameters of the IP 265
0012      :L   FW   18      in slot 2
0013      :T   QW   84
0014      :L   FW   20
0015      :T   QW   86
0016      :
0017      :L   IW   90      Load output parameters of IP 265
0019      :T   FW   22      in slot 3 and store
001A      :L   IW   92      in flag words 22 to 26
001B      :T   FW   24      for processing
001C      :L   IW   94      by the CPU user program
001D      :T   FW   26
001E      :
001F      :L   FW   28      Transfer flag words 28 to 32 to
0020      :T   QW   90      the input parameters of IP 265
0022      :L   FW   30      in slot 3
0023      :T   QW   92
0024      :L   FW   32
0025      :T   QW   94
0026      :BE

```

7 Startup, Loading and Operating States		
7.1	Function Model of a Startup	7 - 1
7.2	Loading	7 - 5
7.2.1	Loading the IP 265 by Means of COM 265 via the External I/O Bus	7 - 5
7.2.2	Loading the IP 265 from the Memory Submodule	7 - 7
7.3	Operating States	7 - 8
7.3.1	Possible Operating States	7 - 8
7.3.2	Useful Operating States for Normal Operation and Transitions ...	7 - 9

Figures		
7-1	Function Model for Startup of the IP 265	7 - 2
7-2	Function Model - IP 265 Changes Operating State	7 - 10
Tables		
7-1	Operator Inputs - Loading Via the I/O Bus	7 - 6
7-2	Behaviour of IP 265 in Case of CPU STOP if IP 265 is in RUN Mode	7 - 9

7 Startup, Loading and Operating States

7.1 Function Model of a Startup

A function model is used to illustrate the basic user inputs required for startup of the IP 265 (Figure 7-1).

You determine the function of the IP 265 in the IP 265 user program. The IP 265 is programmed and tested by means of COM 265 at the programmer/PC.

Note

You only need to plug the standard submodule into the module if you want to use a standard program. In this case, startup is terminated.

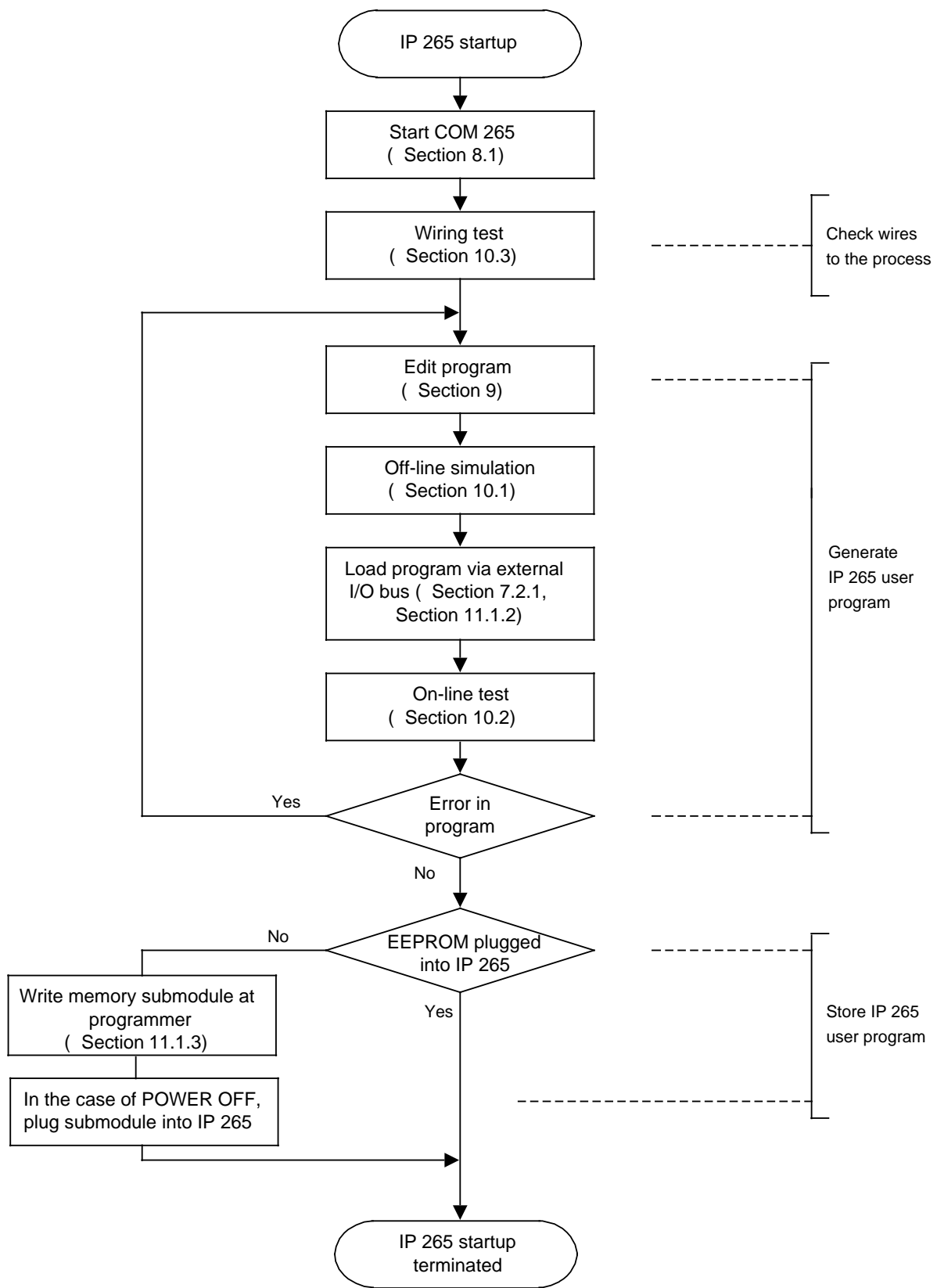


Figure 7-1. Function Model for Startup of the IP 265

Wiring test

COM 265 offers the possibility of checking the wiring between the inputs/outputs of the IP 265 and the sensors and/or actuators of the process interface system for continuity.

COM 265 provides the special "WIRE" user program for the IP 265 to check the wiring.

Program generation

The IP 265 user program is generated in four steps:

- Editing of the IP 265 user program
- Off-line simulation of the IP 265 user program
- Loading of the IP 265 user program into the IP 265
- On-line test of the IP 265 user program

Off-line simulation

With the aid of COM 265, the control of the relevant sub process by the IP 265 user program is tested off-line. Possible program errors can quickly be detected and eliminated without the process interface system being connected.

The user can simulate the IP 265 user program by means of test inputs and display the results of the off-line test. The user specifies input signals for simulation. Changes of the output signals or internal intermediate states in the IP 265 user program can be monitored on the programmer/PC screen.

Loading

For the on-line test, the IP 265 user program must be available in the IP 265.

The IP 265 user program can be loaded into the IP 265

- via the remote I/O bus
or
- from the memory submodule on the module.

Loading of the memory submodule is recommended for normal operation. For on-line testing of IP 265 program execution, loading via the external I/O bus is recommended.

Note

Loading the IP 265 via the external I/O bus is only possible with COM 265.

Note

When using the IP 265 together with the ET 200, the IP 265 user program can only be loaded from the memory submodule.

On-line test

The IP 265 user program is tested on the module in order to check IP 265 program execution. COM 265 provides convenient access to the address area of the IP 265 in the PIQ and PII of the CPU.

With COM 265, the user can "write" and "read"

- the required operating state in the control word of the module
- the status information in the status word of the module
- input parameters of the IP 265 user program
- output parameters of the IP 265 user program

Storing

There are two ways of storing the IP 265 user program on a memory submodule:

- Direct at the programmer (E(E)PROM memory submodule plugged into programmer)
or
- Indirect in the module (EEPROM memory submodule plugged into IP 265. IP 265 is loaded via the remote I/O bus)

Notes

In addition to starting up the IP 265, you must program that section of the CPU user program which controls IP 265 program execution. During generation of the IP 265-relevant section of the CPU user program, some important principles must be taken into account to ensure correct operation of the IP 265.

Programming of the control of IP 265 program execution is therefore treated in a separate section of this manual (Section 6 "Control of the IP 265 by the CPU User Program").

Note

The CPU user program is generated and tested in STEP 5 at the programmer/PC.

Note

The "FORCE VAR" PLC function of the STEP 5 package also provides access to the PIQ and PII of the IP 265.
For startup of the IP 265, IP 265 program execution can also be controlled with the aid of this PLC function.

7.2 Loading

There are two ways of loading the IP 265:

- Loading via the external I/O bus
- Loading from the memory submodule

During both loading procedures, all inputs and outputs of the module are passive. The loading procedure is indicated by high-speed flashing of the STOP LED on the module front and by the relevant status acknowledgements in the status word of the module (Section 4.3.2).

7.2.1 Loading the IP 265 by Means of COM 265 Via the External I/O Bus

Loading via the I/O bus is only possible using COM 265. COM 265 has a special utility: "Loading IP 265 via I/O bus". After calling the utility, COM 265 handles loading of the program data independently via the I/O bus.

Some prerequisites must be fulfilled for the loading procedure:

- The program memory of the programmer/PC contains an IP 265 user program generated by means of COM 265 and compiled into IP 265-relevant data,
- An EEPROM submodule is plugged into the submodule receptacle of the IP 265
or
the submodule receptacle of the IP 265 is empty.

Note

If an EEPROM is plugged into the module, the IP 265 user program is additionally copied onto this memory submodule.

Note

COM 265 evaluates the memory submodule before transferring the program data into the IP 265. If an illegal submodule or an EPROM submodule is plugged into the module, COM 265 outputs an error message.

Operator inputs at the CPU

Since the remote I/O bus is operated via the CPU, user inputs can be made depending on the operating state of the CPU **prior to** inputs with COM 265 (Table 7-1).

User inputs in COM 265

The necessary inputs in COM 265 for initiating the loading procedure are described in Section 11.1.2 "Loading the IP 265 Via the I/O Bus". During the loading procedure, COM 265 coordinates program scanning automatically on the CPU and the IP 265 (Table 7-1).

Table 7-1. Operator Inputs - Loading Via the I/O Bus

Operating State of the CPU	User Handling	Effects
RUN	No further user inputs necessary	CPU user program continues to work, IP 265 program execution is interrupted
STOP via mode selector switch	Mode selector switch of the CPU must be set to RUN prior to loading. After the loading procedure, the CPU must be switched to STOP again.	CPU user program is not processed
STOP caused by error in the CPU user program or operator input on programmer/PC	No other user inputs necessary	CPU is briefly switched to RUN by COM 265. However, CPU user program is not processed.
STOP due to hardware error	Loading only possibly if error has been eliminated	

If the CPU was switched to STOP before beginning the loading procedure, the CPU user program is unlinked during loading of the IP 265 via the I/O bus. A fault during the loading procedure (e.g. POWER OFF or connecting cable between programmer and CPU unplugged) may lead to a loss of the CPU user program. The CPU user program must be reloaded from the memory submodule into the CPU.

Status information

During automatic loading of the IP 265, the control bit is set in the status word. The control bit indicates to the CPU user program that the IP 265 is controlled by COM 265. When all data blocks of the user program have been transmitted to the IP 265, it sets the status bit "IP 265 loaded".

If an error occurs during the loading procedure, the IP 265 enters the STOP state. COM 265 outputs the relevant error message on the programmer/PC screen. The STOP LED flashes. The error flags "group error" and "loading error" additionally appear in the control word.

Note

After a successful loading procedure, the IP 265 enters the STOP state. With the aid of COM 265, the IP 265 can be switched to the RUN state immediately after loading.

Duration of the loading procedure

The IP 265 user program is transmitted within 1000 data cycles. At an average CPU scan time of 20 ms, the loading procedure would last approx. 20 s. If a connected EEPROM memory submodule is overwritten simultaneously, the entire loading procedure lasts approx. 80 s.

7.2.2 Loading the IP 265 from the Memory Submodule

Loading from memory submodule is possible without COM 265.

For the loading procedure, the following prerequisite must be fulfilled:

- A memory submodule containing the IP 265 user program must be plugged into the IP 265. You can use either
 - an EPROM or EEPROM submodule programmed using COM 265
 - or
 - an EPROM submodule with standard program

User handling

POWER ON

Irrespective of whether a valid freely programmable memory submodule or a standard submodule is plugged into the module, the user program is automatically loaded into the IP 265 after **POWER ON**.

Note

After transfer of the program data to the IP 265, the IP 265 evaluates the memory submodule. If an illegal EPROM, an empty EPROM or a memory submodule with invalid program data is plugged into the module, the IP 265 enters the STOP state or hard STOP. The IP 265 indicates these errors in the following ways:

- Error message in the status word (Section 4.3.2; Table 4-3)
- Flashing of the STOP LED (Appendix A.1)

Status information

Automatic loading of the IP 265 from the memory submodule is displayed by the control bit in the status word. If the loading procedure has been successful, the status bit "IP 265 loaded" is set.

Transfer errors during loading, however, set the error flags "group error" and "loading error" in the status word.

Note

After a successful loading procedure, the IP 265 enters the STOP state. It must be explicitly switched to RUN to start program execution. In the control word of the IP 265, the relevant bit ("RUN bit") must be set by the CPU user program for this purpose (Section 4.3.1).

Duration of the loading procedure

The loading procedure takes approx. 1 s.

7.3 Operating States

7.3.1 Possible Operating States

During startup and normal operation, the IP 265 can enter different operating states.

A change of the operating state of the IP 265 can be initiated in the following ways:

- By the user via the control word of the module
 - By programming the CPU user program accordingly (normal operation)
 - With the aid of COM 265 (startup)
 - With the aid of the "FORCE VAR" PLC function of the STEP 5 package (startup)
 - By incorrect operation of the IP 265
- Through the IP 265 and the module environment (COM 265, CPU)
 - CPU entering the STOP state
 - A hardware error (e.g. "wirebreak")

The respective operating state can be evaluated by the CPU user program in the status word of the IP 265. The operating state of the IP 265 is additionally indicated by flashing of the RUN and STOP LEDs.

Note

In Appendix A of the manual, you find two tabular overviews of possible operating states of the IP 265 and their display in the status word and via LED signals:

- LED displays: Appendix A.1
- Error codes in the status word: Appendix A.2

IP 265 response to a CPU STOP

The IP 265 response to a STOP of the CPU is a special case. With the aid of COM 265, you can configure this response (Section 9.2).

You can choose between:

- DISABLED (IP 265 enters STOP state, I/O is passive)
- ENABLED (IP 265 enters STOP state, I/O is active)

Note

If you do not configure the response of the IP 265 separately in the case of a CPU STOP, the IP 265 goes into the STOP state and the inputs and outputs of the IP 265 are automatically switched to the passive state (default setting).

This provides you with the opportunity to continue the sub process controlled by the IP 265 even if the CPU is in the STOP mode.

Table 7-2. Behaviour of IP 265 in Case of CPU STOP if IP 265 is in RUN Mode

Configuration	LEDs	Behaviour
DISABLED (default setting)	STOP LED: Steady light	IP 265 enters STOP state, I/O is passive
ENABLED	RUN LED: Steady light	IP 265 remains in the RUN state, I/O is active

DISABLED:

If the CPU goes into RUN mode again, the IP 265 must be switched explicitly to RUN by a RUN command (control word). Since the PIQ is deleted at a cold restart of the CPU, the input parameters must also be entered into the PIQ in addition to the control word.

ENABLED:

In this case, the IP 265 remains in the RUN mode after a CPU STOP. Since the PIQ is deleted at a cold restart of the CPU, the input parameters must be entered in the PIQ again and the RUN bit in the control word must be set again. Only if the RUN bit is set, are the input parameters processed by the IP 265.

7.3.2 Useful Operating States for Normal Operation and Transitions

The state transition diagram in Figure 7-2 on the next page gives an overview of possible operating states and their transitions during normal operation of the IP 265. Error states which can be initiated through incorrect user inputs and hardware faults are not included in the representation.

Prerequisite for normal operation:

A permissible memory submodule with valid IP 265 user program is inserted in the IP 265.

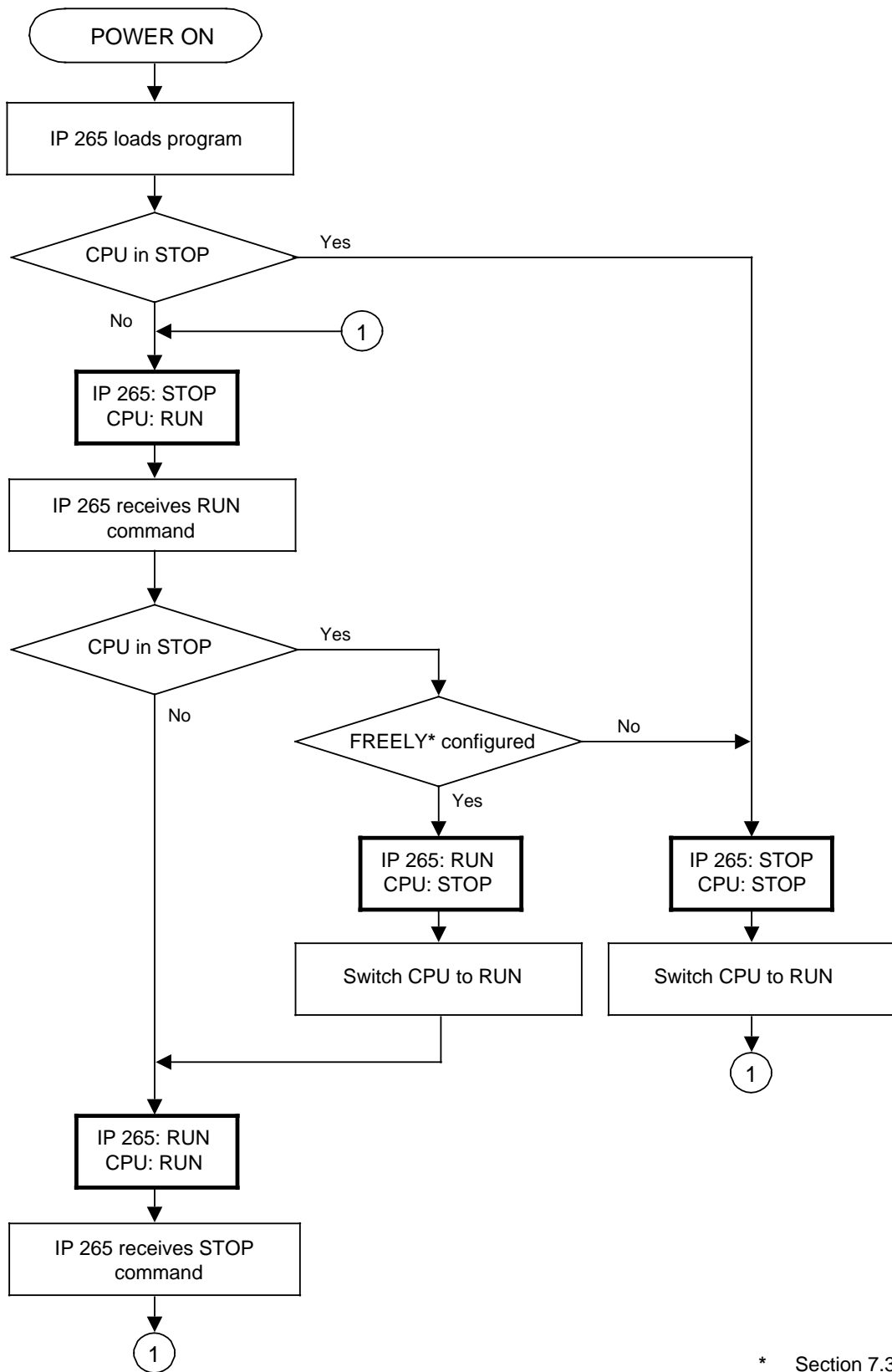


Figure 7-2. Function Model - IP 265 Changes Operating State

8 Fundamentals of COM 265		
8.1	Preparations for Working with COM 265	8 - 2
8.2	COM 265 Screen Forms	8 - 4
8.3	Basic COM 265 Functions in the "Function Selection" Form	8 - 5
8.4	Hierarchical Structure of COM 265	8 - 8
8.5	COM 265 Operator Control Philosophy	8 - 10
8.5.1	Help Forms and Help Windows	8 - 10
8.5.2	Errors and Warnings	8 - 13
8.5.3	Cursor Control in Control System Flowcharts and Input Fields	8 - 15
8.5.4	Key Assignments for Editing Functions	8 - 16

Figures		
8-1	COM 265 "Function Selection" Form; COM 265 Screen Form Segments . .	8 - 4
8-2	Hierarchical Structure of COM 265	8 - 8
8-3	Selecting the "Rename" Function for Files	8 - 9
8-4	Help form: Details on COM 265 On-Screen Form "File Functions" and Function Key Menu	8 - 11
8-5	Help Window: Information on the "Slot number" Input Field in the COM 265 "Defaults" Form	8 - 12
8-6	Example: Error Message Display in the "Compile" Form and Information on Error Recovery	8 - 14
8-7	The COM 265 "Defaults" Form; Slot Number Selection	8 - 15
8-8	Key Assignments for COM 265 Editing Functions	8 - 16
Tables		
8-1	Hardware and Software Requirements for COM 265	8 - 1

8 Fundamentals of COM 265

The COM 265 programming package is a user-friendly software aid for starting up the IP 265. COM 265 is used both to write the IP 265 application program and to execute that program on the IP 265.

Like other SIMATIC COM packages, COM 265 is an autonomous package. Conventional COM packages are used to initialize SIMATIC S5 modules. In contrast, COM 265 has its own **programming language (based on the CSF format used in STEP 5)** for writing and testing a special application for the S5-100's IP 265 module. COM 265 thus has its own utilities and editors, similar to the S5 utilities and STEP 5 packages used for CPU programming.

COM 265 can execute on the PG 730, PG 750 and PG 770 programmers as well as on AT-compatible personal computers (PCs). Hardware and software requirements are listed in Table 8-1.

Table 8-1. Hardware and Software Requirements for COM 265

Hardware requirements for PG and PC	
<ul style="list-style-type: none"> • 80386 CPU • 4 MB RAM • Hard disk with free space totalling 5 Mbytes 	
Software requirements for	
PG	PC
<ul style="list-style-type: none"> • MS-DOS 3.2 • S5-DOS/ST (MS-DOS): V3.2 	<ul style="list-style-type: none"> • MS-DOS 3.2 • STEP 5 basic package for PCs or STEP 5 basic package for the S5-90U

Section 8 begins with descriptions of fundamental COM 265 handling procedures in order to acquaint you with the software package. You will find information on

- COM 265 installation and startup (Section 8.1)
- Screen layouts for interactive screen forms (Section 8.2)
- COM 265 basic functions in the "Function Selection" form (Section 8.3)
- The hierarchical structure of COM 265 (Section 8.4) and
- General COM 265 operator servicing procedures (Section 8.5)

Note

All information on COM 265 operator servicing presented in the following sections relates to its use on the PG 750 programmer. You will find additional information on the use of COM 265 on PCs in Appendix D.

8.1 Preparations for Working with COM 265

Before beginning work with COM 265, you must first make a number of preparations. First, make sure that the following software has been installed on your programmer:

- Operating system: MS-DOS 3.2
- STEP 5 package: S5-DOS/ST (MS-DOS): V3.2

Standard package

COM 265 is provided on two floppy disks in MS-DOS operating system format.

Make backup copies

It is recommended that backup copies be made of all files.

Start the MS-DOS operating system.

Copy the original floppy.

MS-DOS provides the "DISKCOPY" utility for this purpose.

Calling the "DISKCOPY" utility

Call the "DISKCOPY" utility by entering the following:

DISKCOPY A: A: <1>

Please take all other information on the DISKCOPY utility from the appropriate sections of the PG manual.

Install COM 265

Before you can work with COM 265, you must first install a workable version of it on the programmer's hard disk.

Insert the decoder for COM 265 into the programmer's LPT1 interface.

Load the MS-DOS operating system.

Insert COM 265 floppy disk no. 1 into the diskette drive.

Designate the floppy disk drive as the current drive.

Start the installation utility "INSTALL" by entering the following:

INSTALL <1>

You will then be prompted to enter the COM 265 destination path.

Enter the destination drive and a directory name,

e.g. C: COM265 <1>

When prompted, insert COM 265 floppy disk no. 2 in the diskette drive and press <1>.

The installation program displays an appropriate message upon successful installation of COM 265.

Start COM 265

Now start the COM 265 programming package from the hard disk.

Make the directory in which you stored COM 265 into the current directory, e.g.

```
C: <1>  
cd \COM265 <1>
```

Start COM 265 by entering the following:

```
COM265 <1>
```

The COM 265 package's top menu, the so-called "**Function Selection**" menu, is displayed on the programmer screen (Section 8.2).

Note

If less than 540 Kbytes of conventional working memory is available, COM 265 reboots the PG/PC.

Note

When starting COM 265, there must be no active hard disk cache which does **not** execute write accesses to the hard disk **immediately**. Typical examples of such programs are SMARTDRV and PC-CACHE.

If you have installed a program like this in your "autoexec.bat" or "config.sys", you must ensure that all write accesses to the hard disk are executed immediately. See the manuals of the cache programs (e.g. the MS-Windows manual for SMARTDRV) for more detailed information.

Note

Whenever the STEP 5 package is reinstalled, COM 265 must also be reinstalled.

8.2 COM 265 Screen Forms

All COM 265 functions can be invoked by making the appropriate entries in interactive screen forms. All COM 265 interactive screen forms have the same structure. Each screen form is subdivided into four areas, or sections. The structure of the COM 265 screen forms is shown below, using the "Function Selection" form, or menu, as an example.

"Function Selection" form

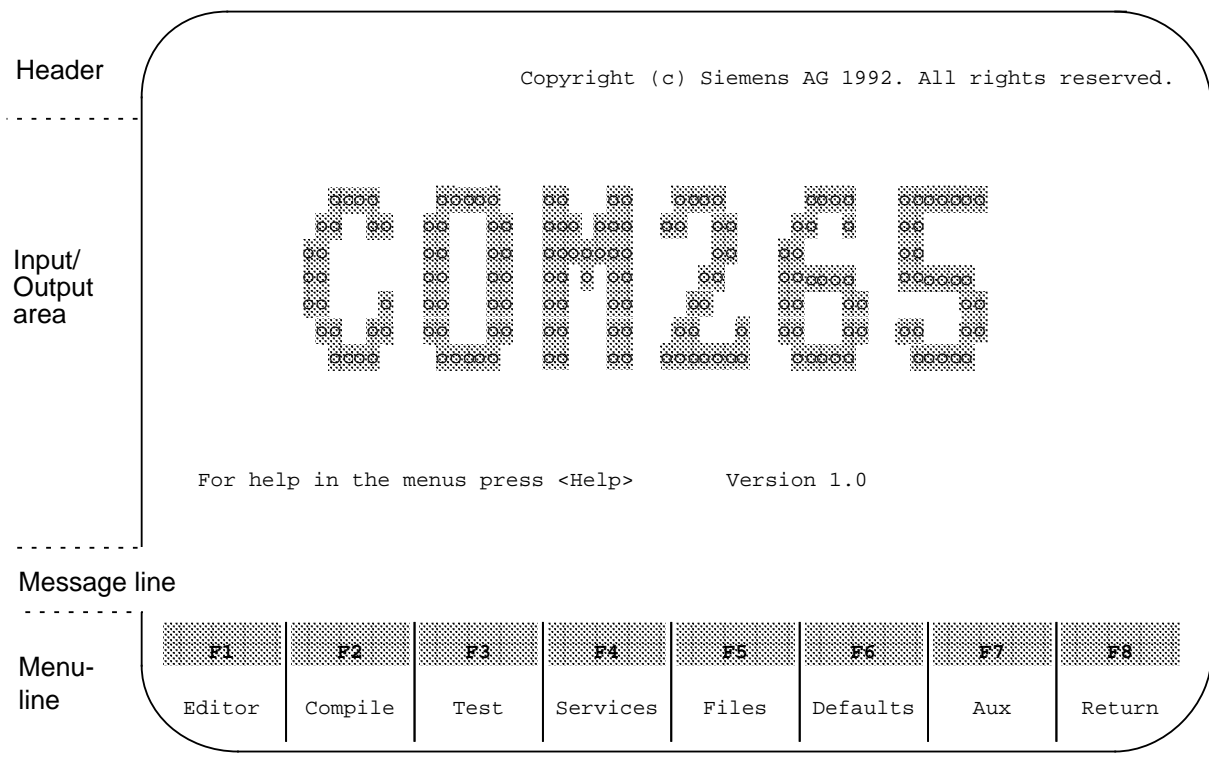


Figure 8-1. COM 265 "Function Selection" Form; COM 265 Screen Form Segments

Header

The header in all COM 265 screen forms comprises one line, and may contain varying information, depending on the operator servicing level and the selected COM 265 function. The user cannot change the header.

Input/output area

The large middle section is the input area in all COM 265 screen forms. This area is used for entering the parameters and program in the PG's program memory, or to display previously stored program data and parameters (output area).

Message line

COM 265 uses the message line to inform you about the progress of an invoked function, operator input errors, or faults/malfunctions.

Menu line

The softkey menu (function keys <F1> to <F8>), which is located at the bottom of the screen, shows the key to press on the programmer keyboard in order to invoke a specific COM 265 function.

8.3 Basic COM 265 Functions in the "Function Selection" Form

The softkey menu in the first COM 265 screen form displayed when COM 265 is started ("Function Selection" form) shows the COM 265 basic functions. You can invoke any one of these functions by pressing the appropriate function key.

The information presented on the next two pages provides a clear, well-ordered overview of COM 265 basic functions.



: Editor

COM 265 allows you to write the IP 265 application program in control system flowchart (CSF) format, which is similar to the CSF format used in STEP 5 (for CPU programming).

- A control system flowchart is a coherent formation of interdependent operations such as logic operations, counter operations, comparison operations, and so on. In addition to the familiar STEP 5 language elements, other language elements may also be linked into in the IP 265 application program.
(Section. 9.3.2)
- In the flowchart, addressing of the operands for the IP 265 application program may be either direct or symbolic.
- In addition to the control system flowchart, you may also enter comments for each segment in the IP 265 application program (segment commentary).
- Background checking tasks are performed during entry of the IP program. For instance, COM 265 reports errors in syntax or semantics on the message line, prompting immediate correction.



: Compile

Before it can be loaded into the IP 265, the IP 265 user program generated on the programmer must first be compiled. (The terms 'application program' and 'user program', as used here, are synonymous).

The IP 265 application program reserves memory space on the IP 265. During the compiler run, checks are made to see whether the application program is translatable and to make sure that there is sufficient room in the IP 265's memory.



: Test

COM 265 provides extensive options for testing the IP 265 user program.

- Using the COM 265 simulator, the user can simulate most of the interconnections in the IP 265 user program off-line, i.e. without PLC or IP 265.
- A second option allows on-line testing of the user program's logic under real-time conditions (conditions for sub control of the overall system) in the PLC using process signals. To enable this form of testing, the IP 265 user program must be loaded into the IP 265.
- Independently of the IP 265 user program, you can also execute a wiring test in order to check the interfaces to the process.



: Services

By pressing the function key assigned to 'Services', the user can invoke the symbolic editor and initiate output of the IP 265 user program to various media.

- The COM 265 operand editor allows you to enter symbolic identifiers for operands in the IP 265 user program.
- Normal operation of an IP 265 requires that the IP 265 user program be stored on an EPROM or EEPROM memory submodule.
- For on-line testing, the IP 265 user program can be loaded into the IP 265 over the external I/O bus.
- To provide hardcopy documentation of the IP 265 user program, program data and commentary can be output to printer.



: Files

The IP 265 user program is stored in the form of a program file. The file functions support program file management.

- The COM 265 file functions can be used to rename, copy or delete the user program.
- In addition, you can use these functions to display the directory of all IP 265 user programs.



: Defaults

Before you begin programming the IP 265, you may define a number of general defaults.

- The defaults include, e.g. specification of a name for the IP 265 user program and of a slot for the IP 265.
- A PLC footer file and a PLC printer file may be activated for output of the IP 265 user program.



: Aux

For purposes of IP 265 control, the user may define the behaviour of the IP 265's 24 V inputs and 24 V outputs.

- Using function keys, you can define the IP 265's response to a CPU STOP and stipulate debouncing of one or more 24 V inputs.

8.4 Hierarchical Structure of COM 265

COM 265 is structured into several operator levels. The top menu, i.e. "Function Selection", represents the highest operator level. Pressing of a function key from <F1> to <F7> takes you to the COM 265 screen form for the basic function you selected. Some basic functions have (one or two) additional screen forms with subfunctions.

Each COM 265 screen form (except for the "Editor" form) can be exited with <F8> ("Return") or <F6> ("Store").

The diagram below illustrates the hierarchical structure of COM 265. Screen forms with subforms have a bold-type border.

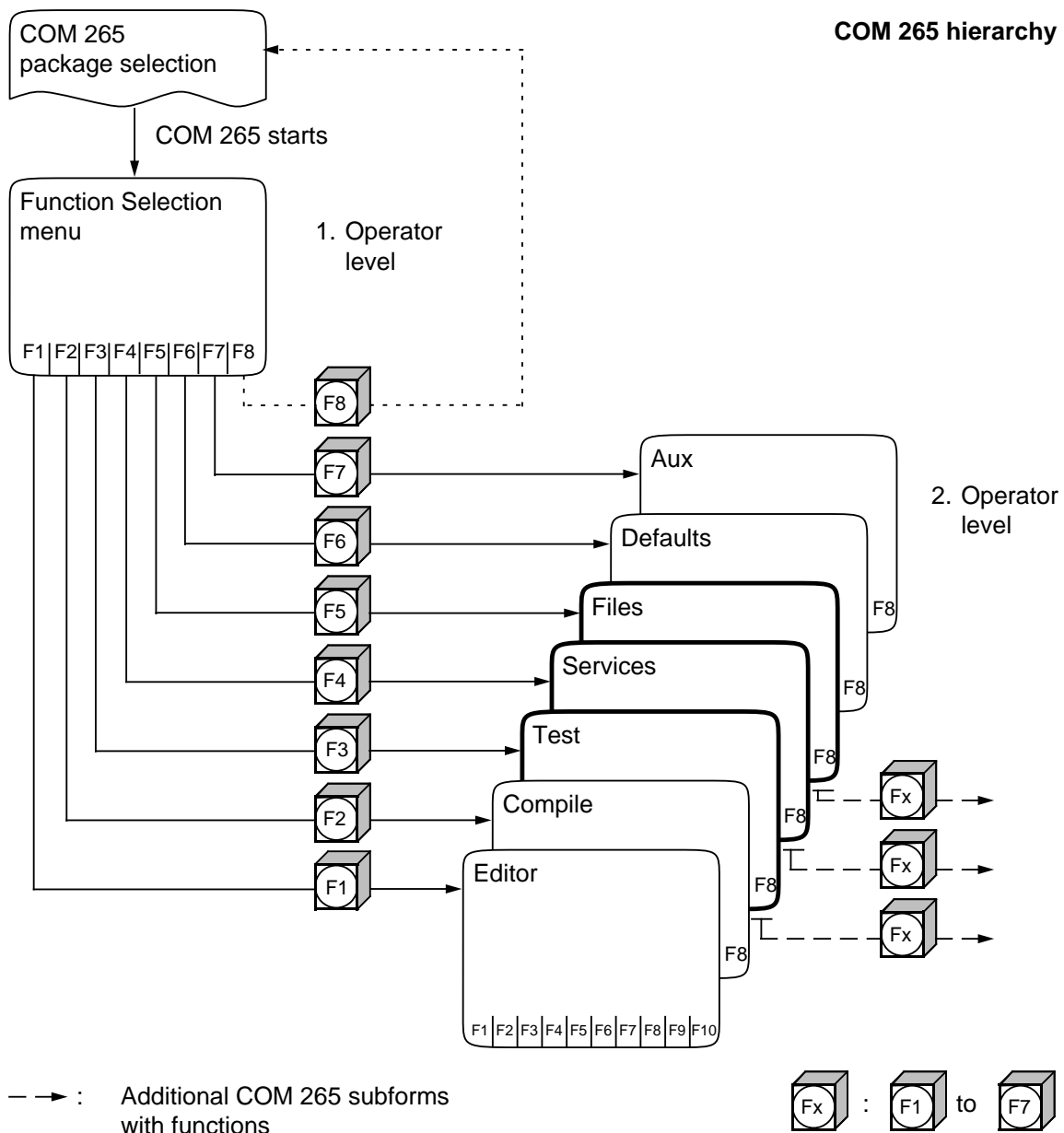


Figure 8-2. Hierarchical Structure of COM 265

The function key menu has the same basic structure for all COM 265 screen forms:

- <F1> to <F7> invoke COM 265 functions or sublevels
- <F8> ("Return") or <F6> ("Store") returns you to the next higher COM 265 screen form (Figure 8.3)

The single exception is the "Editor" form. Seven programmable function keys are not sufficient to cover the numerous language elements in an IP 265 CSF. Ten function keys are reserved in the "Editor" form's first function key menu:

- <F1> to <F9>: Call COM 265 language elements
- <F10> : Call additional function key menu with further COM 265 language elements

The "Editor" form can be exited with <ESC> or <INSERT>.

Example: You want to change the name of an IP 265 user program.
 The path to the COM 265 screen form "Rename" is roughly outlined in Figure 8-3.

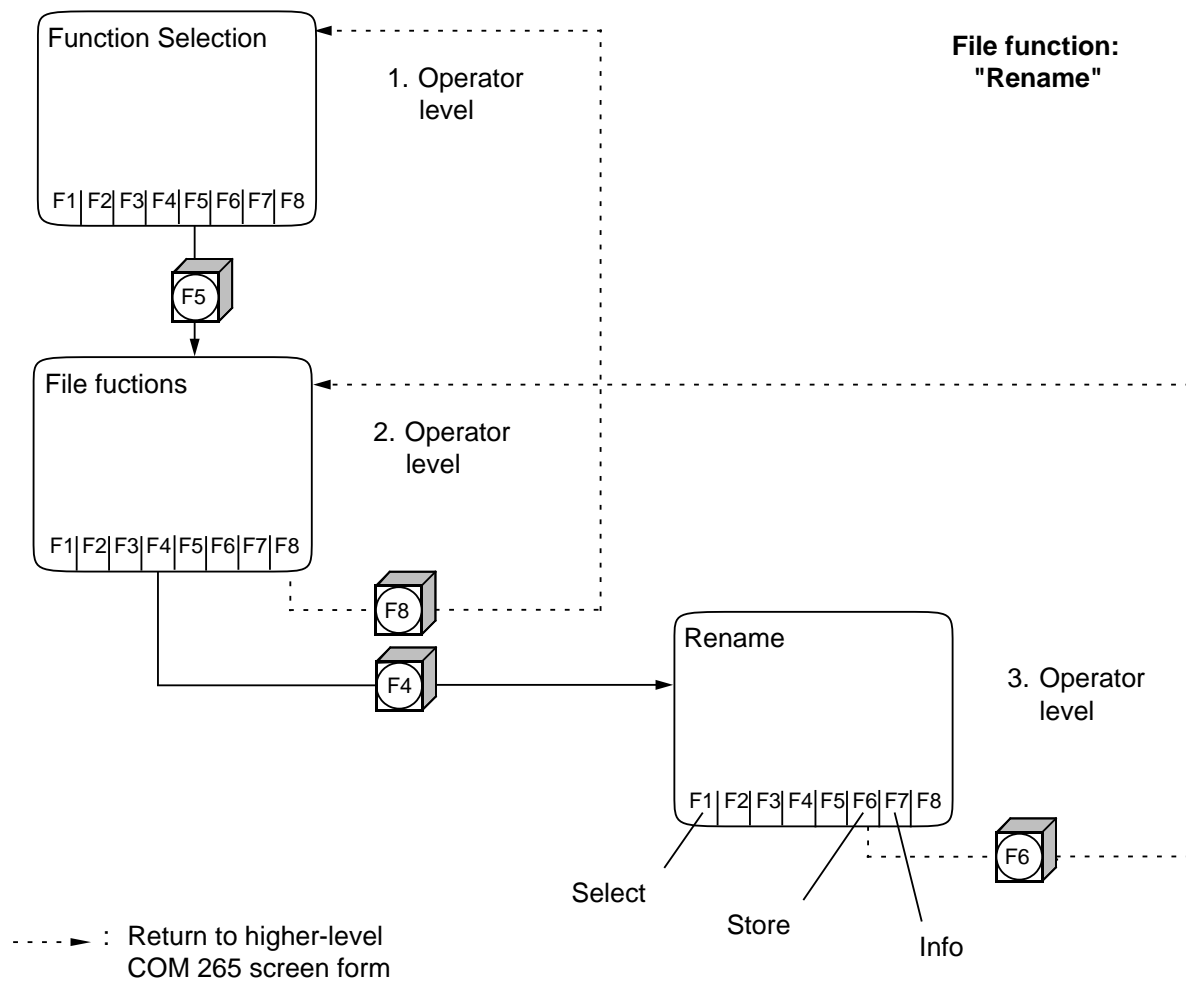


Figure 8-3. Selecting the "Rename" Function for Files

8.5 COM 265 Operator Control Philosophy

8.5.1 Help Forms and Help Windows

The "Help" concept in COM 265 is similar to that used in STEP 5.

Help texts can be displayed in dependence on the selected COM 265 screen form and the current position of the cursor.

COM 265 provides three kinds of Help texts:

Help line:	Help text displayed on the COM 265 message line
Help forms:	Help texts with descriptive information on the current COM 265 screen form and function key menu
Help windows:	Help texts for input fields and Help texts for compiler error messages

Help line: Help text displayed on the COM 265 message line

In the absence of COM 265 operator errors, COM 265 displays information on current functional sequences on the message line (e.g. "Check the IP 265 program ...").

An exception is the COM 265 "Editor" form. In this screen form, an estimation of the load placed on the IP 265 by the IP 265 user program, in %, is displayed on the message line. The **load capacity display** is updated during the entry of COM 265 language elements and operands. This information helps you judge the size and compilability of the IP 265 user program (Section 9.4.1, Section 9.5).

COM 265 outputs one of the following messages on the "Editor" form's message line, depending on the estimated IP 265 load:

- Load xx % < 100 %: "The IP 265 program utilizes approx. xx% of the IP 265's capacity."
- Load xx % 100 %: "IP program utilizes approx. xx% of the IP 265's capacity; possibly uncompileable."
- Load xx % 130 %: "IP program utilizes approx. xx% of the IP 265's capacity; probably uncompileable."

Help forms: Help texts with descriptive information on the current COM 265 screen form and function key menu

You can display a Help form with descriptive information about the current screen form and function key menu by pressing the **<HELP>** key in a COM 265 screen form.

The original screen contents are replaced by the HELP info.

If one screen page is insufficient, you can screen the next page by pressing <INSERT> or <1>.

You can exit the Help form with **<ESC>**. The original screen contents are redisplayed.

Example: Help form: Descriptive information on the current COM 265 screen form (i.e. "Files") and its function key menu.

COM265 file functions

<F1> Copy an IP265 program; drive-to-drive copying is also possible.
 <F2> Erase an IP265 program on the specified drive.
 <F3> Directory of all IP265 programs on the specified drive.
 <F4> Rename an IP265 program on the specified drive.
 <F8> Return to COM265 top menu

Note: At the operating system level, an IP265 program is stored on a file with the name "xxxxxPIP.S5D", but only the "xxxxx" appears in COM265. This portion of the name may also comprise fewer than 5 characters.
 Example: The IP265 program "TEST" is stored under the name "TESTPIP.S5D".

<ESC>: Close window

F1	F2	F3	F4	F5	F6	F7	F8
Transfer	Delete	Dir	Rename				Return

Figure 8-4. Help form: Details on COM 265 On-Screen Form "File Functions" and Function Key Menu

Help window: Help texts for input fields and Help texts for compiler error messages

Press function key <F7> "Info" (if available in the function key menu) for Help texts on COM 265 input fields. A Help window is displayed which provides information on possible and permissible entries in the field at which the cursor is currently positioned.

In contrast to the Help forms for function key menus described above, a Help window for an input field or compiler error message overlays only part of the current screen contents. The input field remains visible.

Only one Help window may be open at any given time.

The Help window must be closed before making an entry in the input field or positioning the cursor to the next input field.

Press <ESC> to close a Help window.

Example: Help window: Information on the "Slot number" input field in COM 265 "Defaults" form.

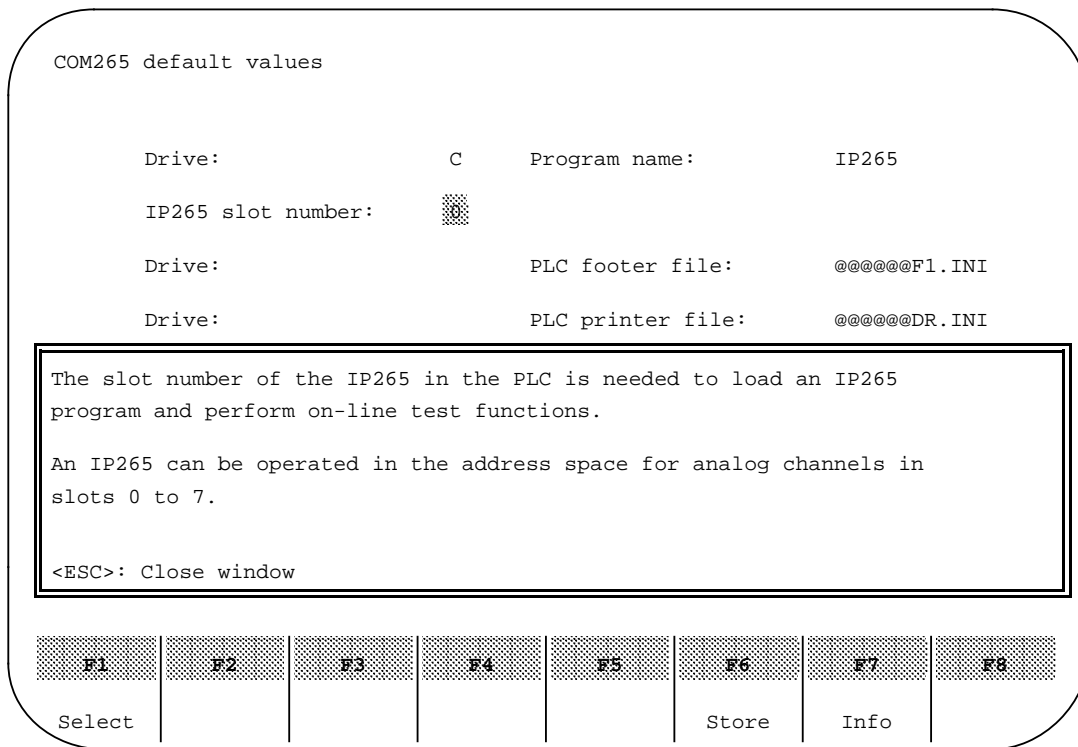


Figure 8-5. Help Window: Information on the "Slot number" Input Field in the COM 265 "Defaults" Form

Note

You can screen more detailed information on the compiler's on-screen error message by pressing <F7> "Info" in the "Compile" screen form. This Help window provides details on the cause of error and recommendations on how to eliminate the error.

8.5.2 Errors and Warnings

The COM 265 concept of errors and warnings is also similar to that used in STEP 5.

COM 265 can detect various types of errors and inform the user by displaying an appropriate message.

COM 265 can detect and report the following types of errors:

- Errors in semantics and syntax in entries to input fields (operator input errors).
- Faults detected during on-line testing and during the execution of service functions (hardware faults).
- Errors detected during compilation of the IP 265 user program (program errors).

COM 265 reports these errors in different ways, depending on their gravity. Some are reported in the form of error messages, some in the form of warnings, and some in the form of 'indications'.

- Error messages and 'notes' are normally displayed on the message line, as in STEP 5.
- Warnings, in contrast, are displayed in a window in the center of the CRT screen (e.g. "Warning! Symbolic declarations have not been stored! Do you want all changes to be lost?"). Warnings must be acknowledged with <ESC> or, if a prompt so indicates, with <ESC> "NO" or < 1 > "YES".

Errors in semantics and syntax in entries to input fields

Entries in COM 265 screen forms, programs and IP 265 user program tests are all subject to specific rules of syntax and semantics.

COM 265 executes semantics and syntax checks on an input text when the user confirms his entry with < 1 >. Entries containing errors are reported (e.g. "Invalid operand address").

Faults detected during on-line testing and during the execution of service functions

COM 265 provides adequate support for on-line testing. On-line functions must fulfill specific hardware prerequisites.

When an on-line function is started, COM 265 checks to make sure that these prerequisites are fulfilled, reporting a fault if this is not the case (e.g. "Wrong or defective memory submodule").

Errors detected during compilation of the IP 265 user program

COM 265 provides adequate support for compiling IP 265 user programs. An IP 265 user program reserves memory space on the IP 265 (Section 9.4.1). When a compiler run is started, COM 265 checks to see whether the IP 265 user program is compilable and whether there is sufficient space in the IP 265's memory.

If the IP 265 user programm cannot be compiled, COM 265 aborts the compilation and displays an error message. You can screen details on the cause of error and recommendations on how to rectify it by pressing <F7> "Info".

Example: Error message displays in the COM 265 "Compile" form and recommendations on how to rectify the errors.

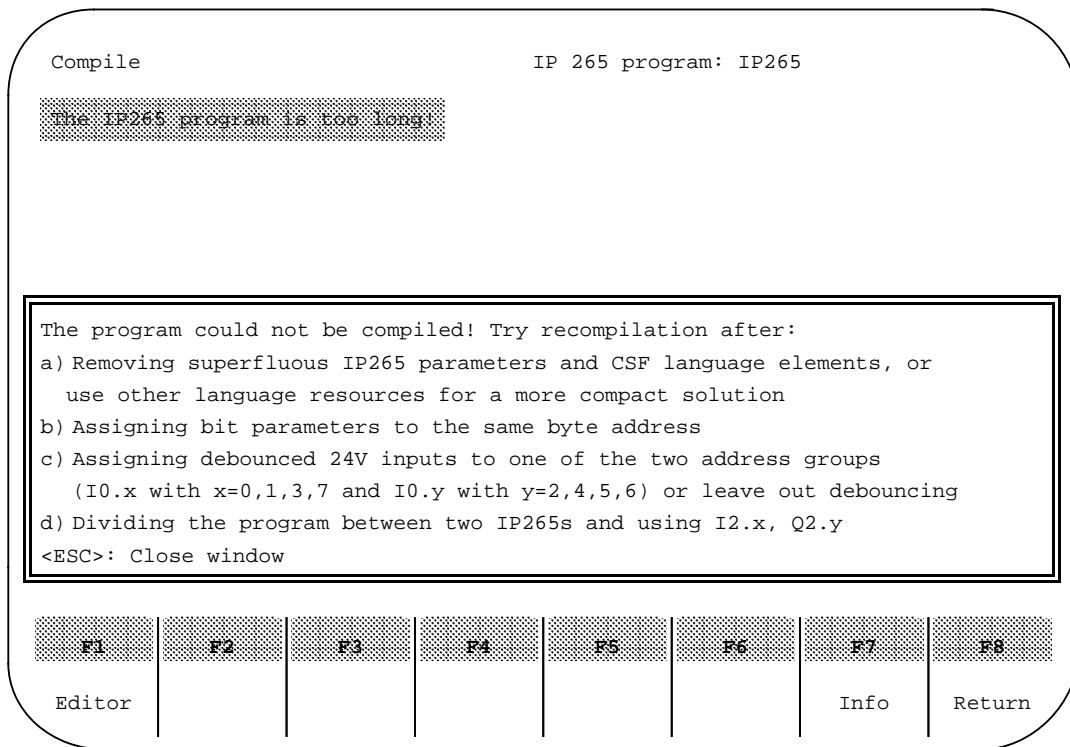


Figure 8-6. Example: Error Message Display in the "Compile" Form and Information on Error Recovery

If the IP 265 user program is free of errors and not too big, the following message is displayed upon successful compilation:

"The program was compiled successfully and utilizes xx % of the IP 265 load capacity."

The value xx % indicates the amount of IP 265 storage taken up by the IP 265 program when it is loaded.

8.5.3 Cursor Control in Control System Flowcharts and Input Fields

All entries in COM 265 screen forms are cursor-supported. The cursor can be moved only to specific positions within each screen form.

Cursor control in and between input fields

- The cursor is used to 'mark' the current input field. When positioned to a new input field, it 'jumps' directly to that field.
- The cursor moves character by character within an input field.
- There are two methods of cursor-supported text entry in input fields
 - The text is entered character by character on the programmer keyboard
 - The text is selected from a window (if available) belonging to the input field. The current window is opened with <F1> "Select".

Example: Selecting the slot number from a window

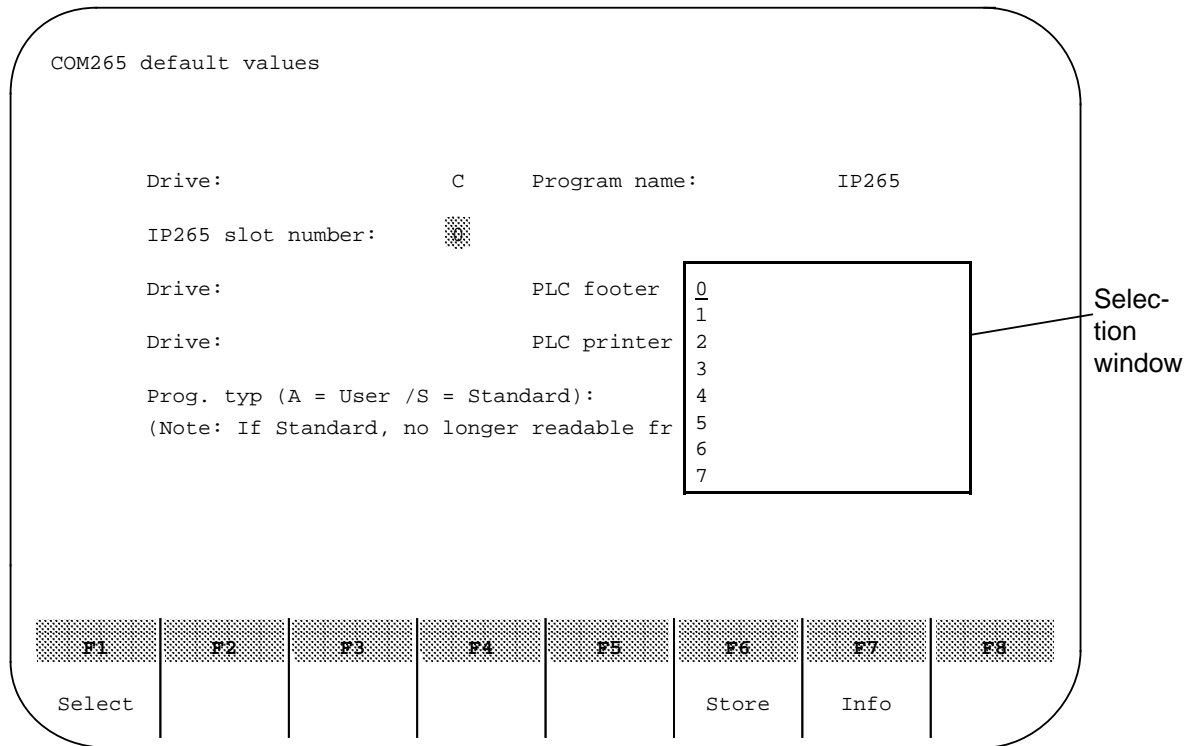


Figure 8-7. The COM 265 "Defaults" Form; Slot Number Selection

Position the cursor to the relevant text line in the selection window and press < 1 >. The selected text appears in the input field.

Cursor control between the language elements in a control system flowchart

In a control system flowchart, the possible cursor positions are based on the line/column grid.

- Using the cursor, mark the grid positions in the control system flowchart at which actions such as "Insert language element" or "Invert input" are to be carried out. Connecting lines, language elements and marginal objects can be marked in the control system flowchart.

Section 8.5.4 provides information on the keys on the programmer keyboard that can be used for cursor control.

8.5.4 Key Assignments for Editing Functions

All COM 265 editing functions can be invoked via keys on the programmer keyboard's numeric keypad or via special function keys.

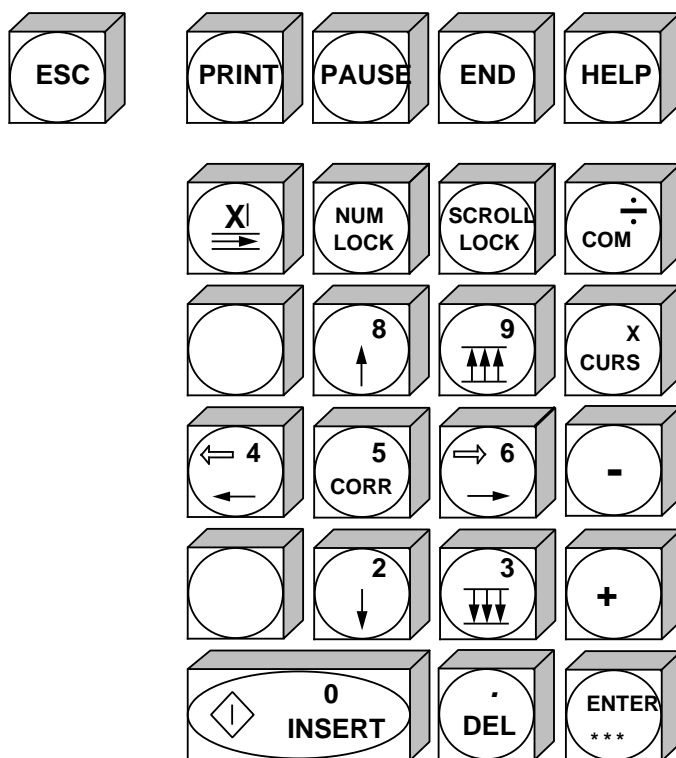
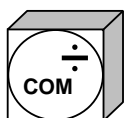


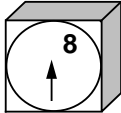
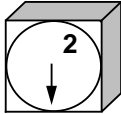
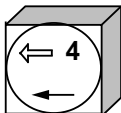
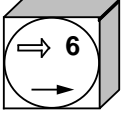
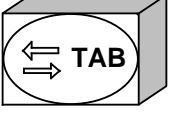
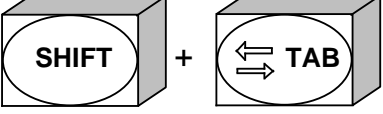
Figure 8-8. Key Assignments for COM 265 Editing Functions

Inserting comments:

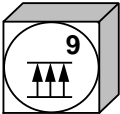
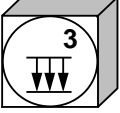
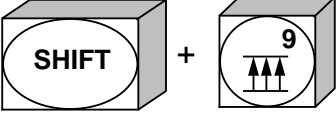
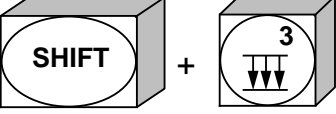


- : Commentary key
- Press <COM> once only: Enter segment name
- Press <COM> twice in succession: Enter segment commentary

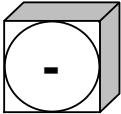
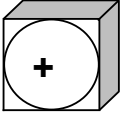
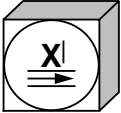
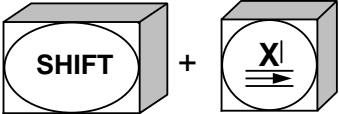
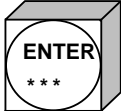
Cursor control:

-  : Move cursor up one line in input field;
Move cursor up based on grid in control system flowchart
-  : Move cursor down one line in input field;
Move cursor down based on grid in control system flowchart
-  : Move cursor one character to the left in input field;
Move cursor to the left based on grid in control system flowchart
-  : Move cursor one character to the right in input field;
Move cursor to the right based on grid in control system flowchart
-  : Move cursor to next input field at right;
Move cursor from input bar to output bar
-  : Move cursor to next input field at left;
Move cursor from output bar to input bar

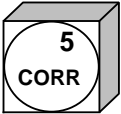
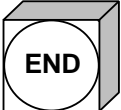
"Move" screen:

-  : Scroll up one line
-  : Scroll down one line
-  : Page down
-  : Page up

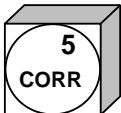
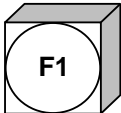
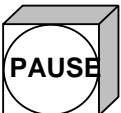
Segment handling:

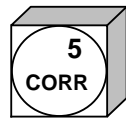
-  : Page to previous segment
-  : Page to next segment
-  : Insert segment in front of on-screen segment. The segment numbers of all the following segments will be incremented by one.
-  : Delete the on-screen segment. The segment numbers of all segments that follow the deleted segment will be decremented by one.
-  : Enter new segment at end of program.

Go from "Simulator" to "Editor" and vice versa:

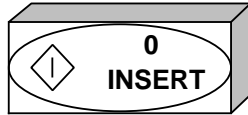
-  : Correction key
Direct change from simulator to the control system flow-chart editor of COM 265 without having to open the "Select Function" menu.
-  : Change back to the simulator. The changes in CSF can also be transferred (user is prompted).

Go from "Compiler" to "Editor" or vice versa:

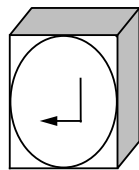
-  or  : Correction key or function key <F1> "Editor"
Go direct from compiler to COM 265 control system flowchart editor without opening the "Function Selection" menu
-  : Return to compiler, with or without storing changes made in CSF (prompt).

Go from "Editor" to "Assignment line":

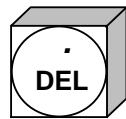
: Go direct from the control system flowchart editor to a section of the symbolic editor (assignment line) without exiting the "Editor" form (Section 9.4.3).

Store functions:

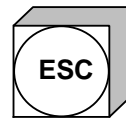
: Store (confirm) key



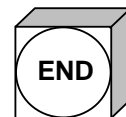
: Return key

Delete function:

: Delete key
Deletes characters, language elements or inserted inputs of a language element marked by cursor.

Special functions not on numeric keypad:

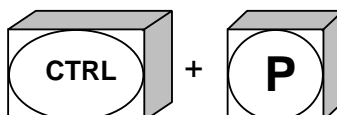
: Abort key (<ESC> key)
Depending on the status, the user may be asked if he really wants to abort



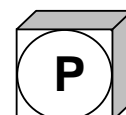
: Go from control system flowchart editor to simulator. Changes made in the CSF may be stored.



: Information on the current function key layout in the on-screen COM 265 form.



+



: Print CSF segment currently on simulator screen. In text format, the entire IP 265 user program is output. In the symbols editor, all operands are listed.

9 Programming the IP 265 with COM 265

9.1	Defaults	9 - 1
9.2	Configuring the IP 265 Response	9 - 4
9.3	COM 265 Language Description	9 - 6
9.3.1	Operands in a Control System Flowchart	9 - 8
9.3.2	Language Elements in a Control System Flowchart	9 - 11
9.4	Enter IP 265 User Program	9 - 27
9.4.1	Rules and Recommendations for Programming	9 - 27
9.4.2	Entries in the Local IP 265 Assignment List	9 - 30
9.4.3	Entering a Control System Flowchart	9 - 33
9.4.4	Entering the Segment Name and Segment Commentary	9 - 46
9.5	Compiling the IP 265 User Program	9 - 48
9.5.1	IP 265 User Program Load on the IP 265	9 - 49
9.5.2	What if there is an IP 265 Overload?	9 - 51

Figures

9-1	"Defaults" Form; Defaults for COM 265	9 - 1
9-2	"Configure" Form; Configuring Data for the IP 265	9 - 4
9-3	"Editor" Form; IP 265 User Program Segments	9 - 6
9-4	Basic Structure of "Timer" Language Elements	9 - 18
9-5	Basic Structure of "Counter" Language Elements	9 - 23
9-6	Basic Structure of "Comparison Operations" Language Elements	9 - 25
9-7	Connectors in the IP 265 User Program	9 - 26
9-8	Distributor	9 - 26
9-9	Connection Element	9 - 26
9-10	"Symbols" Form; Enter Operand Assignment List	9 - 31
9-11	"Editor" Form; Enter the Control System Flowchart	9 - 34
9-12	"Editor" Form; Enter Connectors	9 - 39
9-13	"Editor" Form; Enter Assignment Line	9 - 40
9-14	"Commentary" Form; Enter Segment Commentary	9 - 47
9-15	"Compile" Form; Placing COM 265 Language Elements	9 - 48

Tables

9-1	Marginal Objects in a Control System Flowchart	9 - 9
9-2	Binary Logic Operations	9 - 12
9-3	"Set/Reset operations"	9 - 13
9-4	Binary scaler	9 - 14
9-5	Edge flag	9 - 15
9-6	Delay	9 - 16
9-7	Example: Using the "Delay" Language Element	9 - 17
9-8	Timer Operations	9 - 19
9-9	Menu Line "Clock-Pulse Generator"	9 - 22
9-10	Clock-Pulse Generator	9 - 22
9-11	Counter Operations	9 - 24
9-12	Comparison Operations	9 - 25
9-13	IP 265 Load Effected by COM 265 Language Elements and their Interconnection	9 - 50

9 Programming the IP 265 with COM 265

The following sections cover procedures for programming the IP 265 with COM 265.

It has been assumed that you have made the preparations for working with COM 265 discussed in Section 8.1.

COM 265 provides Help forms and Help windows (Section 8.5.1). You can screen information on the selected COM 265 form by pressing <HELP> and <F7> "Info".

9.1 Defaults

Before starting to program, you should set a number of defaults relating to all COM 265 functions.

"Defaults" form

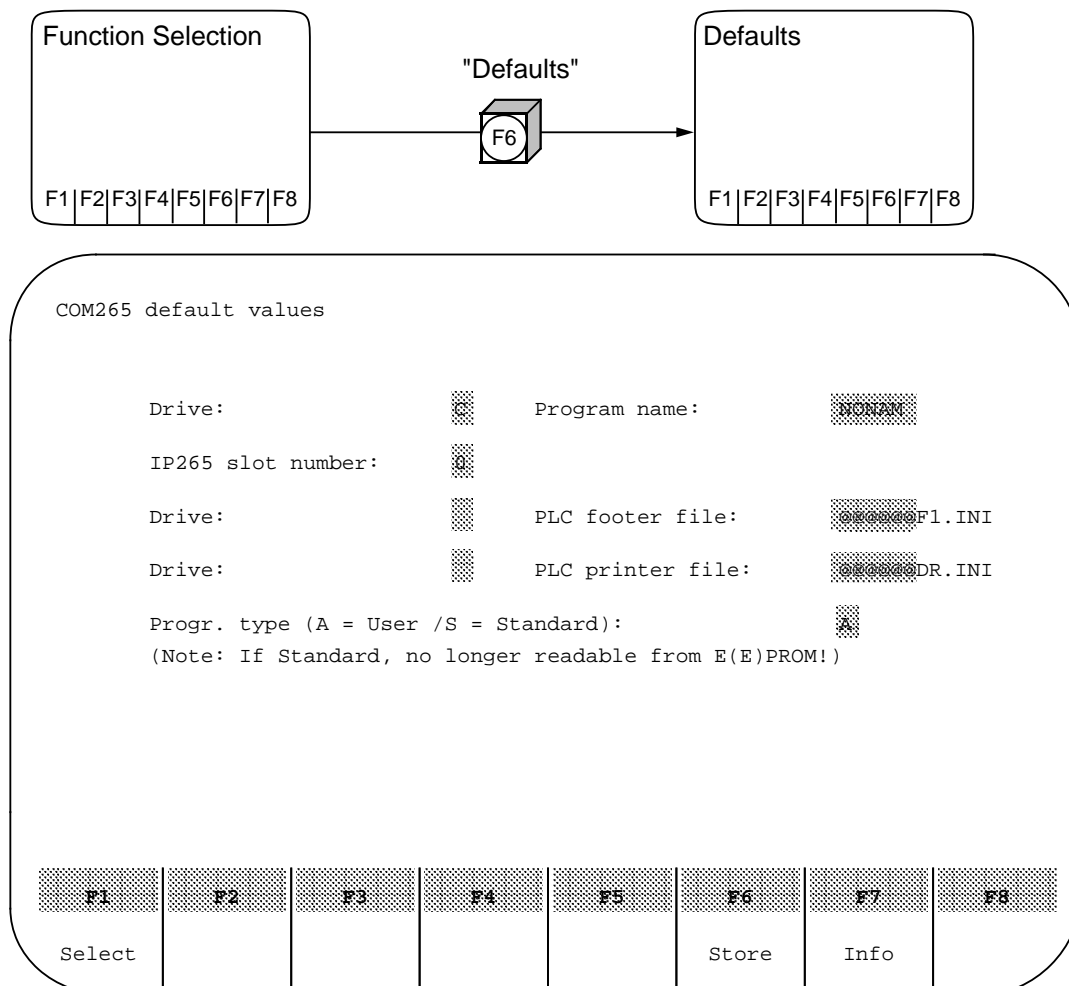


Figure 9-1. "Defaults" Form; Defaults for COM 265

Possible entries in the input fields of the COM265 "Defaults" form:

- Program name:
The IP 265 user program is stored as a program file. For file management purposes, each IP 265 user program must be assigned a name.

In the case of a new program, you can enter a program name comprising no more than five characters in this field. If, for instance, an IP 265 user program has the name "TEST", COM 265 stores it in program memory under the name "TESTPIP.S5D".

If you want to process an existing program, select the name of that program from the selection window for this input field.

- IP265 slot number:
COM 265 needs the IP 265's slot number for the on-line functions "Load via I/O bus" and "On-line test".
- PLC footer file:
COM 265 has access to the PLC's footer file. A footer is a text which the programmer has the printer add at the bottom of each printed page. Choose the desired footer file from the selection window for this input field.
- PLC printer file:
COM 265 has access to the PLC's printer file. The printer file contains the printer parameters for the printout (e.g. character size and number of lines per page).

Note

The PLC footer file and the PLC printer file are external files, and can be programmed only with the STEP 5 software, not with COM 265. In order for COM 265 to have access to both PLC files, they must be stored in (copied to) the same directory as COM 265.

- Drive:
Before calling one of the above mentioned files, you must tell COM 265 which drive it is on.
- Program type:
You may yourself specify whether an IP 265 user program is to be stored on a memory sub-module as user program or as standard program.
In contrast to a user program, a standard program
 - cannot be read back out of the memory submodule,
 - is therefore not modifiable, and
 - cannot be copied.

The specifications in the "Defaults" form can either be entered on the keyboard or chosen from the selection window for the relevant input field. You can screen additional information on the input fields and various editing options by pressing <F7> "Info".

Note

COM 265 stores all entries in the "Defaults" form on **one** file. On a COM 265 cold restart, the last entry made in each input field is regarded as the default for that field, and as such is transferred to the "Defaults" form. All COM 265 functions are subsequently based on the **current** default values.

Example:

Re-editing the program name

Set the cursor to the "Program name" input field (with <TAB>).
Enter the required program name (e.g. "IP265") on the programmer keyboard.
Confirm your entry with <1> or <INSERT>. (Or press <ESC> to revoke your entry).

Picking the slot number from the selection window

Position the cursor (with <TAB>) to the "IP265 slot number" input field (unless it is already set to this field!).
Open the selection window for this input field by pressing <F1> "Select".
Set the cursor to the relevant text line in the selection window.
Transfer the selected slot number to the input field by pressing <1> or <INSERT> (or revoke your selection with <ESC>).

Forward the two default values to program memory by pressing <F6> "Store".

9.2 Configuring the IP 265 Response

Additional defaults can be defined for IP 265 control. With COM 265, you can specify

- Debouncing for each of the eight 24 V inputs
- The response of the IP 265 inputs and outputs in the event of a CPU STOP

"Configure" form

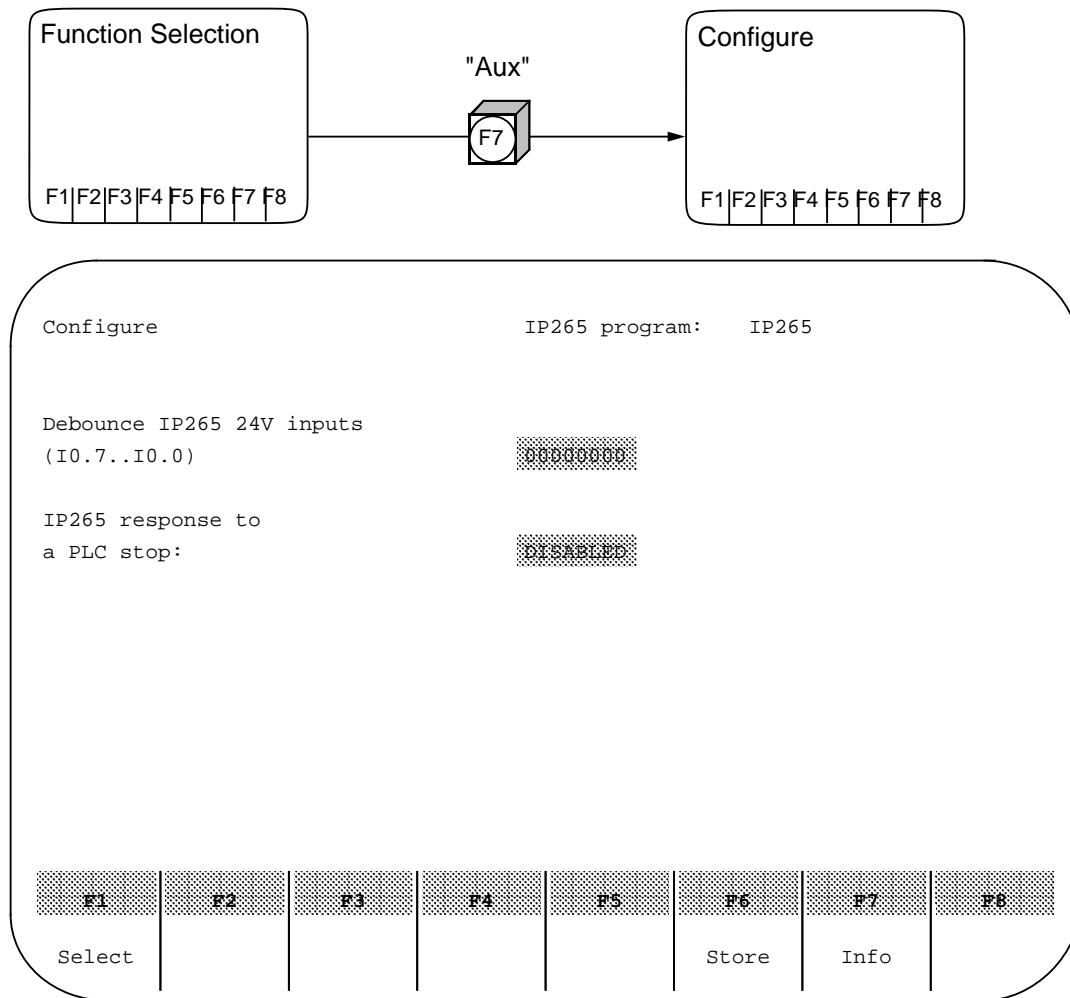


Figure 9-2. "Configure" Form; Configuring Data for the IP 265

The first time the "Configure" form is invoked, the input fields for configuring data contain the default values for the IP 265; these may be confirmed or changed.

You may make your entries in the input fields by typing them in character by character on the keyboard, or by picking them from the relevant selection windows. You can screen information on input fields and editing options by pressing <F7> "Info".

Note

The configuring data belongs to the IP 265 user program, and is stored together with that program in program memory. On a cold COM 265 restart, the configuring data for the selected IP 265 user program is inserted in the appropriate fields in the COM 265 "Configure" form.

Note

You must make any necessary modifications in the "Configure" form prior to compiling the IP 265 user program, at the latest.

Example:**Debouncing input I 0.0**

Position the cursor to the "Debounce 24 V inputs (I0.7..I0.0)" input field.
Move the cursor character by character to the right in the input field until you reach the rightmost character.
Enter a "1" on the programmer keyboard.
Confirm your entry with < 1 > or <INSERT> (or revoke it with <ESC>).

Configuring "active" response from the IP 265 inputs/outputs in the event of a CPU STOP

Set the cursor to the "IP265 response to a PLC STOP" input field (unless it is already positioned at this field!).
Open the associated selection window with <F1>.
Position the cursor to the text line "ENABLED".
Confirm your selection with < 1 > or <INSERT> (or revoke it with <ESC>).

Store the two items of configuring data in program memory by pressing <F6>.

If all entries are successful, input I 0.0 appears in the control system flowchart marked with an "at" sign (@).

9.3 COM 265 Language Description

The IP 265 user program is written with the aid of COM 265 in a control system flowchart (CSF) format similar to that used in STEP 5 (for programming SIMATIC CPUs).

In contrast to the CPU user program, the IP 265 user program is **not** subdivided into blocks which have to be called during program scanning. The IP 265 user program's control system flowchart is comprised only of segments. Segments are editing aids, and do not prescribe the order in which the IP 265 is to process them.

A segment in an IP 265 user program

You program the segments yourself with the COM 265 editor. The editor numbers the segments. Each segment consists of

- a logic field with COM 265 language elements (such as logic operations, timer operations, counter operations, and so on) and connecting lines
- one input and one output bar (margin bars) with operands.

"Editor" form

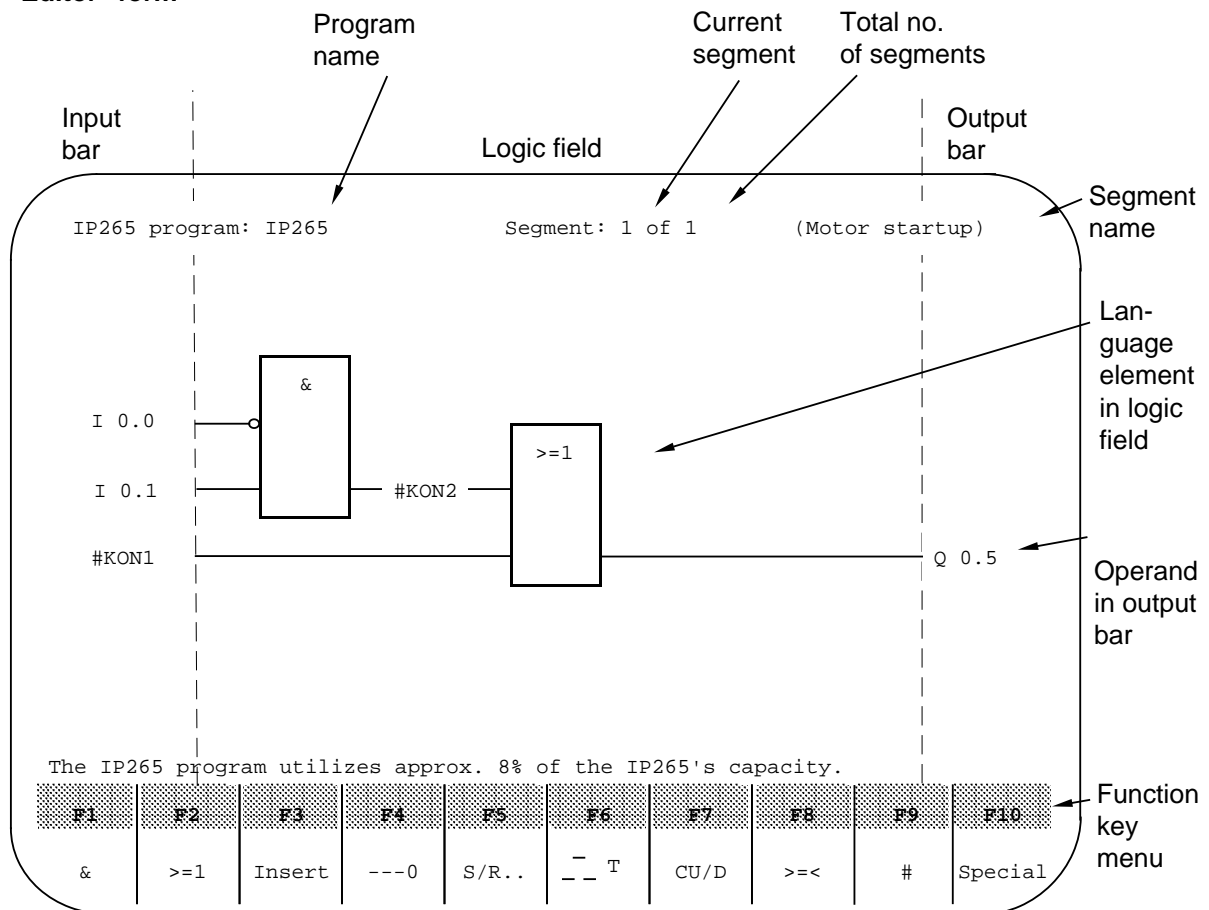


Figure 9-3. "Editor" Form; IP 265 User Program Segments

Operands

- Control system flowchart operands can be placed only in the segments' margin bars.
- The only exception is the connectors, which can also be programmed in the logic field between COM 265 language elements.

Binary connections

- Binary connections between the language elements can be inverted. Inversion is identified symbolically by a "squiggle" at the language element's input (at output when connecting line leads to marginal bar).
- Binary connections between the language elements can be supplemented through the use of connectors, to achieve structuring of the IP 265 user program.

Connectors

Connectors can be used both in the margin bars and in the logic field of a control system flowchart. They are used to branch from connections in and between the segments.

- In addition to a # as language symbol, each connector must be assigned a symbolic name (of no more than 24 characters).
- Connectors can be "written to" only once in the IP 265 user program, but they can be "read" as often as required and at different locations in the program.

Note

IP 265 connectors can also carry non-binary variables. The data type of the connectors is always determined by the connection.

Comments

In addition to the segment name, segment comments and operand comments may be input in order to better document the IP 265 user program. You can program

- A comment for each segment in the IP 265 user program (segment comment)
- A comment for each operand in the control system flowchart (operand comment)

9.3.1 Operands in a Control System Flowchart

The operands in a control system flowchart are placed in "margin bars" (with the exception of connectors, which may also be placed in the logic field).

The following operands are allowed in a control system flowchart's input bar:

- IP 265 inputs
- Input parameters
- Connectors
- Constants

The following operands are permitted in a control system flowchart's output bar:

- IP 265 outputs
- Output parameters
- Connectors

Input and output parameters enable communication between the IP 265 user program and the CPU user program. You will find detailed information on the interchange of parameters between CPU and IP 265 in the manual in Section 4.3.3 "Parameters of the IP 265 User Program" and in Section 6 "Control of the IP 265 by the CPU User Program".

The operands are written with direct or symbolic addresses (identifier) in the margin bars.

Note

In the control system flowchart, symbolic identifiers have the prefix "-", the exception being the parameters. Direct and symbolic identifiers for input and output parameters in a control system flowchart have the prefix "=".

In this Product Manual, symbolic operands are covered in detail in a separate section (i.e. Section 9.4.2).

The IP 265 operand area is more restricted than the CPU operand area. The table below provides an overview of the operands which may be used in the margin bars of a control system flowchart.

Table 9-1. Marginal Objects in a Control System Flowchart

Operand	Data type	Direct address in the IP 265 user program	Access mode in the IP 265 user program
24 V input	Bit	I 0.y ; y (0 to 7)	Read access
24 V output	Bit	Q 0.y ; y (0 to 7)	Write access
5 V differential input (RS 422)*	Bit	I 1.y ; y (0 to 2)	Read access
Extended I/O*	Bit	x 2.y ; x=E or A (exclusively I or Q) y (0 to 7)	x=I: Read access x=Q: Write access
Input parameter	Bit	I x.y ; x (66 to 71) y (0 to 7)	Read access
	10-bit count	IW x ; x (66, 68, 70)	Read access
	10-bit count	IW x ; x (66, 68, 70)	Read access
Output parameter	Bit	Q x.y ; x (66 to 71) y (0 to 7)	Write access
	10-bit count	QW x ; x (66, 68, 70)	Write access
Count	10-bit count	x ; x (0 to 999) KC x ; x (0 to 999) KH y ; y (0 to 3E7) KF+x ; x (0 to 999)	Read access
Time value	10-bit time value	KT x.y ; x (0 to 999) y (0, 1, 2) ".y" optional	Read access
Connectors	Depending on inter-connection	#	Read access, Write access (read/write when connector is between two language elements in the logic field)

* The IP 265 user program may include either operands for differential inputs or for extended I/Os, but not for both.

Constants

The IP 265 user program may include:

- Bit constants
- Count constants
- Time constants

Bit constants:

You may use the two bit constants "0" and "1". Bit constants are used, for example, to specify the response of the COM 265 language element "Clockpulse generator".

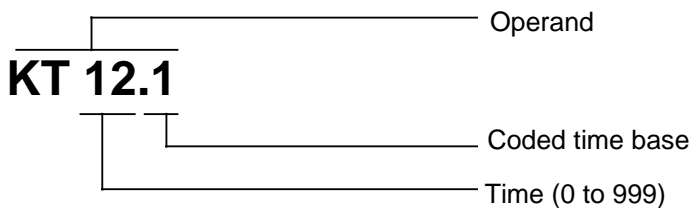
Count constants:

Numeric constants can be used to set counters (count 0 to 999) and comparators, and may be represented in

- Hexadecimal: e.g.: KH 0C or
- Decimal: e.g.: 12, KF+12, KC 12

Time constants:

Preset values for timers may be specified in the IP 265 user program as constants. Time constants must begin with "KT".



Key for the time base:

Time base	No postfix	.0	.1	.2
Time factor	0.001 s	0.01 s	0.1 s	1 s

Parameters

The use of input and output parameters in the IP 265 user program is described in detail in Section 4.3 "Output and Input Data" and in Section 5 "Addressing".

9.3.2 Language Elements in a Control System Flowchart

The COM 265 language elements and their representation in a control system flowchart are very closely based on STEP 5 (which is used to program SIMATIC CPUs).

In addition to the familiar STEP 5 language elements, a number of new language elements have been implemented for programming the IP 265:

- The majority of COM 265 language elements are relevant logic operations which can also be programmed with STEP 5. In CSFs used to program CPUs, they do not exist as autonomous language elements, but rather are generated by using flags and interconnecting segments.
- There are also a number of COM 265 language elements which cannot be programmed in STEP 5, and which would not serve a practical purpose in that language, such as high-frequency clockpulse generators and timers with time values in the millisecond range.

The additional language elements, as well as other process-related differences between the CSFs used for programming the IP 265 and the CSFs used for programming CPUs are emphasized in the text by the use of **italics**.

Note

Edge-sensitive inputs of the COM 265 language elements are explicitly marked with " —> " (positive edge sensitivity) or " —< " (negative edge sensitivity).

The COM 265 language scope includes the following language elements:

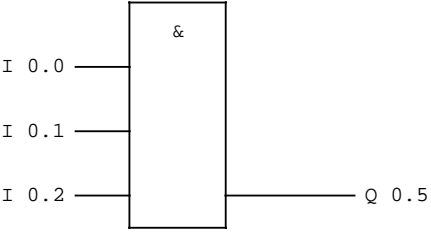
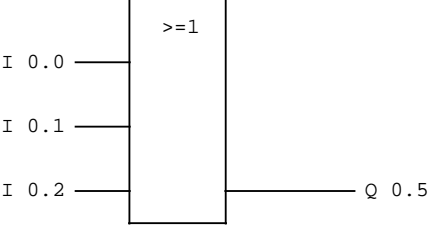
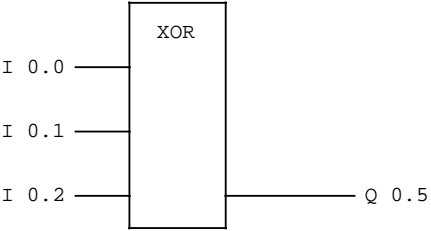
- Binary logic operations (Page 9-12)
- Set/Reset operations (Page 9-13 f.)
- Timer operations (Page 9-18 f.)
- Clockpulse generator (Page 9-22)
- Counter operations (Page 9-23 f.)
- Comparison operations (Page 9-25)
- Connectors (Page 9-26)
- Distributors (Page 9-26)
- Connection element (Page 9-26)

Note

The IP 265 has no retentive memory areas.

Binary logic operations (&, >=1, XOR)

Table 9-2. Binary Logic Operations

Binary logic operation	CSF (example)
<p>AND operation "&": This operation queries whether or not several conditions are all true. An AND operation can evaluate from 2 to 8 inputs.</p> <p>Example: Output Q 0.5 is "1" when all three inputs are "1". The output is "0" when at least one input is "0".</p>	
<p>OR operation ">=1": This operation queries whether one of two (or more) conditions is true. The OR operation can query from 2 to 8 inputs.</p> <p>Example: Output Q 0.5 is "1" when at least one of the inputs is "1". The output is "0" when all of the inputs are "0".</p>	
<p>XOR operation "XOR": This operation queries whether an odd number of inputs are "1". XOR can query from 2 to 8 inputs.</p> <p>Example: Output Q 0.5 is "1" when an odd number of inputs are "1". The output is "0" when an even number of inputs are "1".</p>	

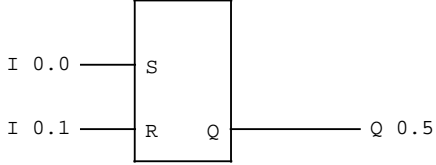
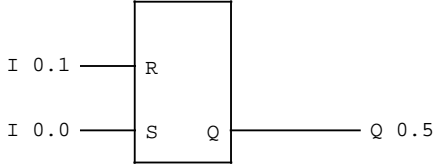
In CSFs for IP 265 programming, the XOR operation is an explicit language element. In a STEP 5 CSF, this operation is not an autonomous language element.

Possible options for logic operations (Section 9.4.3):

- Each of the three language elements may encompass as many as 8 inputs.
- Each of the three language elements may encompass as few as 2 inputs.
- Each of the 2 to 8 inputs may be inverted.
- *Outputs may be inverted when the output of the relevant language element leads directly to the output bar of the control system flowchart.*
- Each of the three binary logic operations can be converted into another binary logic operation or a Set/Reset operation (prerequisite: binary logic operation with two inputs).

Set/Reset operations (S/R..)

Table 9-3. "Set/Reset operations"

Set/Reset operation	CSF (example)
<p>Set/Reset operation for latching signal output (reset dominant) "S": The language element has one Set input (S), one Reset input (R) and one output (Q). The Reset input is dominant.</p> <p>Example: If inputs I 0.0 and I 0.1 are both "1", the operation is reset dominant. Output Q 0.5 is "0".</p>	
<p>Set/Reset operation for latching signal output (set dominant) "R": The language element has one Set input (S), one Reset input (R) and one output (Q). The Set input is dominant.</p> <p>Example: If inputs I 0.0 and I 0.1 are both "1", the operation is set dominant. Output Q 0.5 is "1".</p>	

Note

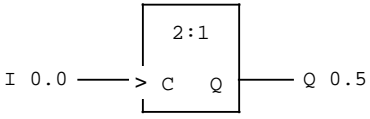
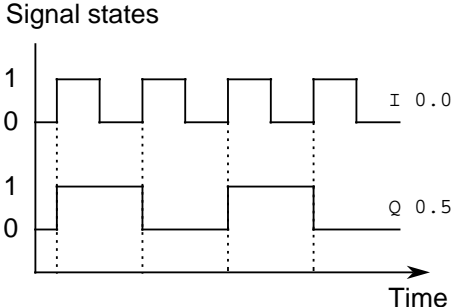
The Set/Reset operations are not assigned a flag number in the IP 265 user program.

Possible options for Set/Reset operations:

- Each input can be inverted.
- *Outputs can be inverted when the output of the relevant language element leads immediately to the output bar of the control system flowchart.*
- Each of the two Set/Reset operations can be converted into its opposite operation or into a binary logic operation.

Binary scaler

Table 9-4. Binary scaler

Set/Reset operation	CSF (example)
<p>Binary scaler "2:1": The language element has one binary input (C) and one binary output (Q). Each rising input signal edge inverts the signal state of the output, effecting a division of the frequency by two.</p> <p>Example: A change in the signal state of input I 0.0 from "0" to "1" inverts the signal state of output Q 0.5, effectively halving the frequency at Q 0.5.</p>	
	<p style="text-align: center;">Timing diagram</p> 

In IP 265 control system flowcharts, the binary scaler is an explicit language element. In STEP 5 (SIMATIC CPU programming), a binary scaler is implemented by the use of flags and the interconnection of several language elements (2 segments).

Possible options for binary scalars:

- The input can be inverted.
- The output can be inverted when it leads immediately to the control system flowchart's output bar.
- The binary scaler "2:1" can be converted into the type conforming language elements "EF", "D-FF" and "Clock pulse" (refer to information on the pages following).

Edge flag (trigger)

Table 9-5. Edge flag

Set/Reset operation	CSF (example)
<p>Edge flag "EF": The language element has one binary input (C) and one binary output (Q). Each rising input signal edge sets the output to "1" for the duration of one internal clock pulse (128 kHz). Each falling input signal edge can set output Q to "1" for the duration of one internal clock pulse (128 kHz) when you invert input C.</p> <p>Example: When input I 0.0 goes from "0" to "1", connector KON1 is set for a period of 7.8 μs.</p>	<div data-bbox="938 434 1299 539" style="text-align: center;"> </div> <div data-bbox="1027 573 1230 607" style="text-align: center;"> <p>Timing diagram</p> </div> <div data-bbox="906 645 1353 1021"> <p>Signal states</p> <p style="text-align: center;">$t: 1/128 \text{ kHz} = 7.8 \mu\text{s}$</p> </div>

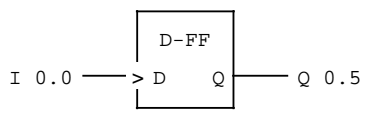
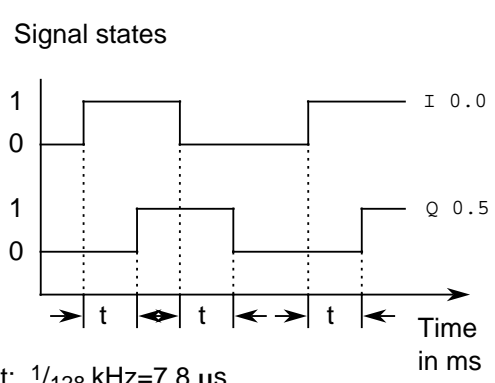
In IP 265 CSFs, the edge flag is an explicit language element. In STEP 5 (SIMATIC CPU programming), an edge flag is implemented by the use of flags and the interconnection of several language elements.

Possible options for edge flags:

- The input can be inverted.
- The output can be inverted when it leads immediately to the control system flowchart's output bar.
- Edge flag "EF" can be converted into type-conforming language elements "2:1", "D-FF" and "Clock pulse".

Delay

Table 9-6. Delay

Set/Reset function	CSF (example)
<p>Delay "D-FF": The language element has one binary input (D) and one binary output (Q). The input signal is delayed by an interval of one internal clock pulse (128 kHz).</p> <p>Example: Each time the signal state of input I 0.0 changes from "0" to "1" or from "1" to "0", the signal state of output Q 0.5 also changes, but this change is delayed by 7.8 μs.</p>	
	<p>Timing diagram</p>  <p>Signal states</p> <p>Time in ms</p> <p>$t: 1/128 \text{ kHz} = 7.8 \mu\text{s}$</p>

STEP 5 (which is used for programming SIMATIC CPUs) does not have this language element; since CPU programs are scanned sequentially, it is not needed.

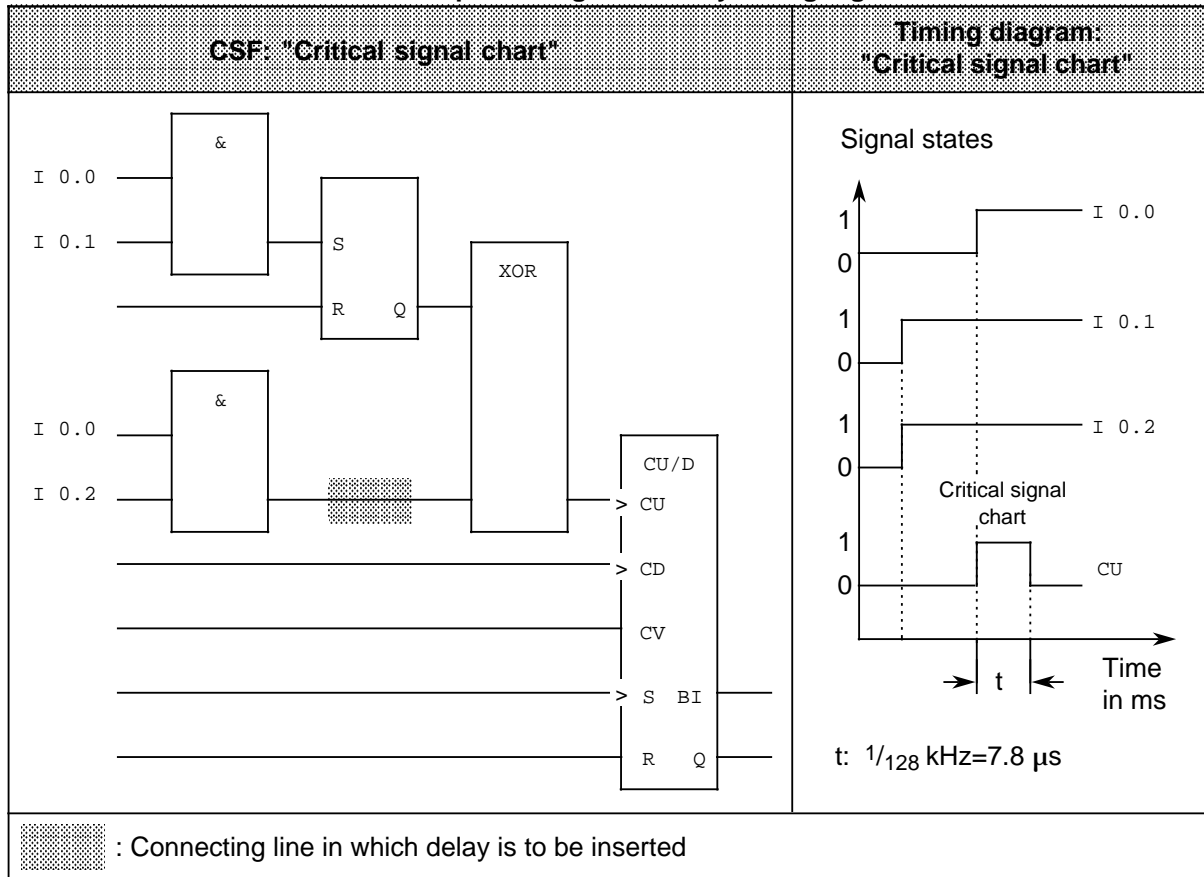
Possible options for delays:

- The input can be inverted.
- The output can be inverted when it leads immediately to the control system flowchart's output bar.
- The delay can be converted into type-conforming language elements "2:1", "EF" and "Clock pulse".

The "D-FF" delay function is sometimes required e.g. for synchronizing latching language elements. The significance of this COM 265 language element is illustrated in an example on the following page.

Example: Critical signal chart due to latching language element (Set/Reset operation) in a segment of the IP 265 user program

Table 9-7. Example: Using the "Delay" Language Element



- Purpose of the partial segment (Table 9-7):
 The counter is always to count up when one of the two "AND" conditions is true. If both of the "AND" conditions are true at one and the same instant, or if neither of the "AND" conditions is true, no counting pulse is to be generated at counting input CU (prerequisite: the RS flipflop's Reset input (R) is "0").
- Development of the critical signal chart:
 The information presented in Section 4-1 of the Product Manual has shown us that a latching language element (such as an RS flipflop) in a segment generates an additional delay of 7.8 μs . In our example, this means that, should the conditions for both "AND" operations be true at the same time, the result of the first "AND" operation would be incorporated into the "XOR" operation 7.8 μs after the result of the second "AND" operation. For a period of 7.8 μs , therefore, it would be as though the results of the two "AND" operations were not identical, even though a positive signal change would take place at the same instant at inputs I 0.1 and I 0.2 (Table 9-7: Timing diagram). Erroneously, a brief positive edge would be generated at the counter input.
- Averting the critical signal chart:
 Placing the "Delay" language element in the second circuit of the segment would synchronize the signal via the RS flipflop.

Timer operations (_ _ T)

The timer operations in an IP 265 user program are very similar to those in a CPU user program, with two exceptions:

- The IP 265 timer operations have no time element numbers.
- Current times for IP 265 timer operations cannot be read out.
- The time base for IP 265 timer operations differs from that used in STEP 5 (CPU programming).

All IP 265 timer operations have the same structure:

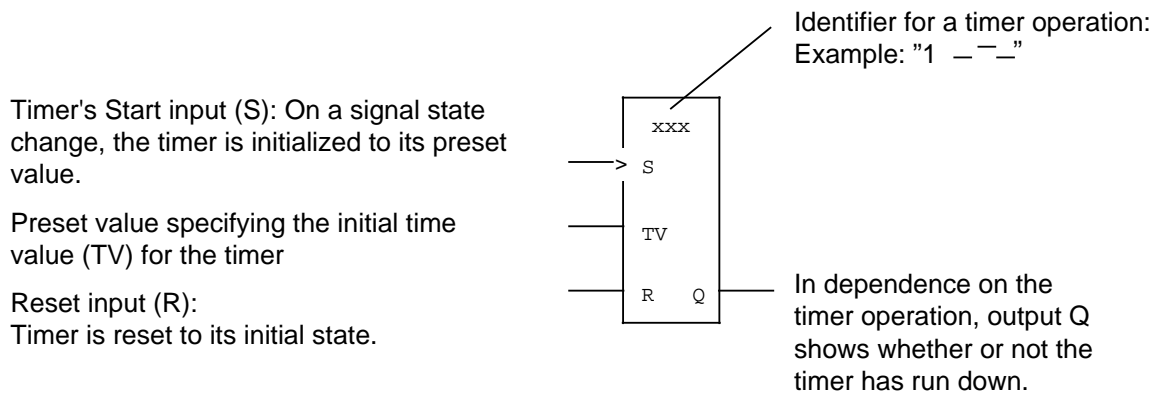
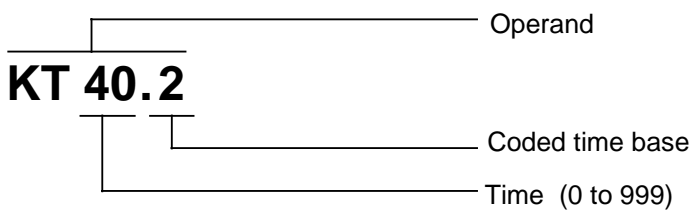


Figure 9-4. Basic Structure of "Timer" Language Elements

The preset value is defined via a time constant or parameter.

The time constant is defined by specifying a time base (coded) and a time:



Key for time base:

Time base	No postfix	.0	.1	.2
Time factor	0.001 s	0.01 s	0.1 s	1 s

Parameters which define times must be written with symbolic identifiers in the IP 265 user program. The initialization of timers is discussed in detail in Section 9.4.2 of the Product Manual. Parameter initialization via the CPU user program is covered in Section 6.

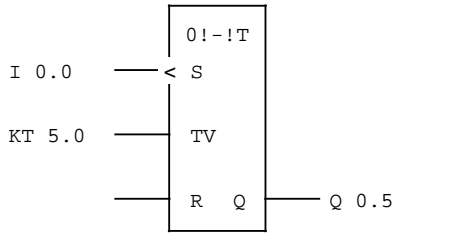
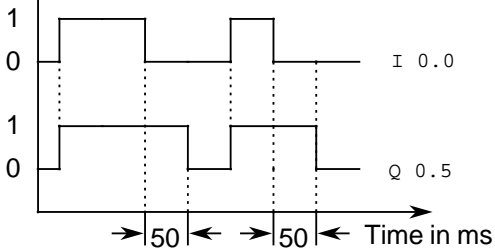
Table 9-8. Timer Operations

Timer operation	CSF (example)
<p>Starting a pulse timer "SI": The timer is started each time the signal at the Start input changes from "0" to "1", and runs until the Start signal is "0", the time has expired or the timer is reset. Output Q is "1" as long as the timer is running, the Start signal is "1" and the Reset signal is "0".</p> <p>Example: Output Q 0.5 is set when input I 0.0 goes from "0" to "1", and is to remain set for no more than 50 ms.</p>	<div data-bbox="906 360 1286 589" data-label="Diagram"> </div> <div data-bbox="1027 629 1230 663" data-label="Section-Header"> <p>Timing diagram</p> </div> <div data-bbox="906 685 1366 1014" data-label="Figure"> </div>
<p>Starting an extended pulse timer "SE": The timer is started (or retriggered) each time the signal at the Start input goes from "0" to "1", and runs until the set time has expired or the timer is reset. A "0" at the Start input does not affect the timer. Output Q is "1" as long as the timer is running and the Reset signal is "0".</p> <p>Example: Output Q 0.5 is set for 50 ms as soon as input I 0.0 goes from "0" to "1".</p>	<div data-bbox="906 1059 1299 1288" data-label="Diagram"> </div> <div data-bbox="1027 1328 1230 1361" data-label="Section-Header"> <p>Timing diagram</p> </div> <div data-bbox="906 1384 1382 1713" data-label="Figure"> </div>

Table 9-8. Timer Operations (continued)

Timer operations	CSF (example)
<p>Starting a delay timer "SD": Each signal change from "0" to "1" at the Start input starts the timer, which runs until the Start signal is "0", the set time has expired or the timer is reset. Output Q goes to "1" when the timer has run down, the Start signal is "1" and the Reset signal is "0".</p> <p>Example: Output Q 0.5 is set 50 ms after input I 0.0, and remains set as long as the input is "1".</p>	<div data-bbox="906 371 1286 600" style="border: 1px solid black; padding: 5px;"> <p style="text-align: center;">T!-!0</p> <p>I 0.0 —> S</p> <p>KT 5.0 — TV</p> <p>— R Q — Q 0.5</p> </div> <div data-bbox="1027 640 1230 674" style="text-align: center; background-color: #cccccc;"> <p>Timing diagram</p> </div> <div data-bbox="906 696 1374 1032"> <p style="text-align: center;">Signal states</p> <p style="text-align: right;">I 0.0</p> <p style="text-align: right;">Q 0.5</p> <p style="text-align: right;">Time in ms</p> </div>
<p>Starting a latching ON delay timer "SS": The timer is started each time the signal at the Start input changes from "0" to "1", and runs until the set time has expired or the timer is reset. If a rising signal edge is detected at the Start input before the set time has expired, the timer is restarted. Output Q goes to "1" when the timer has run down and the Reset signal is "0".</p> <p>Example: Output 0.5 is set 50 ms after input I 0.0 is set.</p>	<div data-bbox="906 1088 1286 1317" style="border: 1px solid black; padding: 5px;"> <p style="text-align: center;">T!-!S</p> <p>I 0.0 —> S</p> <p>KT 5.0 — TV</p> <p>I 0.1 — R Q — Q 0.5</p> </div> <div data-bbox="1027 1357 1230 1391" style="text-align: center; background-color: #cccccc;"> <p>Timing diagram</p> </div> <div data-bbox="906 1413 1374 1805"> <p style="text-align: center;">Signal states</p> <p style="text-align: right;">I 0.0</p> <p style="text-align: right;">I 0.1</p> <p style="text-align: right;">Q 0.5</p> <p style="text-align: right;">Time in ms</p> </div>

Table 9-8. Timer Operations (continued)

Timer operations	CSF (example)
<p>Starting an OFF delay timer "SF": The timer is started each time the signal at the Start input goes from "1" to "0", and runs until the Start signal is "1", the set time has expired or the timer is reset. Output Q goes to "1" when the timer is running or the Start signal is "1" and the Reset signal is "0".</p> <p>Example: Output 0.5 is set to zero 50 ms after input 0.0.</p>	<div style="text-align: center;">  </div> <div style="text-align: center; border: 1px solid black; padding: 5px; margin-top: 10px;"> Timing diagram </div> <div style="text-align: center;"> <p>Signal states</p>  </div>

Note

Time values become imprecise as they approach the time base. For this reason, you should always choose the smallest possible time base.

Possible options for timer operations:

- Start and Reset inputs can be inverted.
- *The timer functions' output can be inverted when it leads immediately to the output bar.*
- Each of the five timer functions can be converted into one of the other remaining four.

Clockpulse generator (Clock)

The IP 265 user program may include clock-pulse generators with variable clock frequencies. You may choose between 8 different frequencies. Select the required frequency in the menu line for the relevant language element.

Table 9-9. Menu Line "Clock-Pulse Generator"

Function key	F1	F2	F3	F4	F5	F6	F7	F8
Clock frequency	1 Hz	10 Hz	100 Hz	500 Hz	1 kHz	2 kHz	16 kHz	64 kHz

Table 9-10. Clock-Pulse Generator

Clock-pulse generator	CSF (example)
<p>Clock-pulse generator "Clock pulse": The language element has one binary input (EN) and one binary output (Q). When the signal state at the input is "1", a clock pulse is generated at the output. The selected frequency is in the upper portion of the language element.</p> <p>Example: Output 0.5 is clocked at 1 kHz when input 0.0 is "1".</p>	<div style="text-align: center;"> </div> <p style="text-align: center;">Timing diagram</p> <div style="text-align: center;"> </div>

In IP 265 user programs, the clock-pulse generator is a separate language element. In STEP 5 (programming language for SIMATIC CPUs), there is no such language element.

Possible options for clock-pulse generators:

- The input can be inverted.
- The output can be inverted when it leads immediately to the control system flowchart's output bar.
- The clock-pulse generator "Clock pulse" can be converted into type-conforming language elements "EF", "2:1" and "D-FF".

Counter operations (CU/D)

You can count both up and down with the IP 265. These two counter operations in an IP 265 user program are largely identical to those used for CPU programming, with two exceptions:

- The current count is available only as binary value.
- The IP 265 counter functions have no counting element numbers.

Up counters and down counters are combined into a **single** language element:

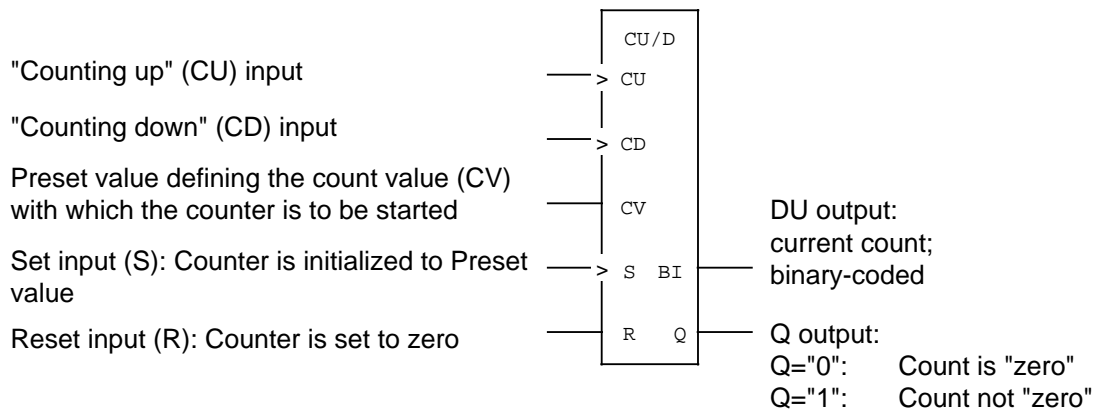


Figure 9-5. Basic Structure of "Counter" Language Elements

The Preset value must be an integer constant or parameter. Parameter entry and deletion procedures are discussed in detail in Section 6. Section 13.1 "Sample Programs" contains programming examples for counters.

Integer constants as Preset values for counters can be represented as follows:

- hexadecimal: e.g.: KH 0C
- decimal: e.g.: 12, KF+12, KC 12

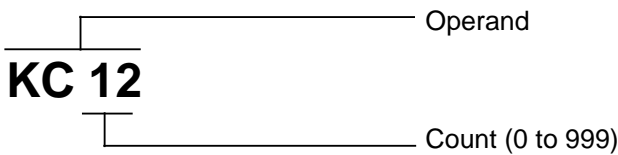


Table 9-11. Counter Operations

Counter operation	CSF (example)
<p>Up counter "CU": Each change from "0" to "1" at the CU input increments the count by 1. The counter is not incremented once it has reached the upper limit value (999).</p>	
<p>Down counter "CD": Each change from "0" to "1" at the CD input decrements the count by 1. The counter is not decremented once it has reached the lower limit value (0).</p> <p>Example: The counter is set to 7 when input 0.1 is set. Output 0.5 is now "1". Each time input 0.0 is set (down counting), the count is decremented by 1. The output is set to "0" when the count is "0".</p>	<p style="text-align: center;">Timing diagram</p>

Possible options for counter functions:

- The following inputs and outputs for counter functions may be inverted: S, R, CU, CD, Q.

Comparison operations (>=<)

Comparison operations make it possible to compare the BI output of a counter function (Table 9-11) with a constant or input parameter in the IP 265 program.

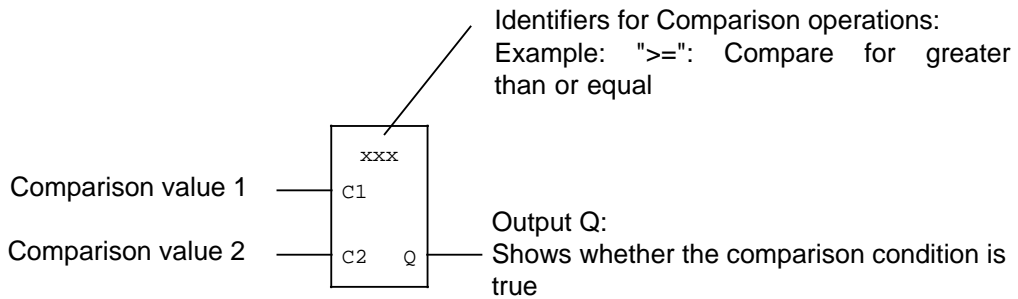


Figure 9-6. Basic Structure of "Comparison Operations" Language Elements

Table 9-12. Comparison Operations

Comparison operation	CSF (example)
Compare for equal "=": C1 and C2 are compared for equal.	<p>Example: Compare for equal</p> <p>Two counts are compared. If they are equal, output 0.5 is set.</p>
Compare for not equal "><": C1 and C2 are compared for not equal.	
Compare for greater than or equal ">=": Comparison to see whether C1 is greater than or equal to C2.	
Compare for greater than ">": omparison to see whether C1 is greater than C2.	
Compare for less than or equal "<=": Comparison to see whether C1 is less than or equal to C2.	
Compare for less than "<": Comparison to see whether C1 is less than C2.	

Possible options for Comparison functions:

- Output Q can be inverted if the Comparison function's output leads direct to the output bar.
- Each Comparison function can be converted into another Compare function.

Connectors (# or <#>)

Connectors can be used in IP 265 application programs both in the margin bars and in the logic field. Connectors make it possible to branch signal leads within a segment and between the segments in the application program.

Each connector must be assigned a symbolic name. The symbolic name for a connector may not exceed 24 (ASCII) characters.

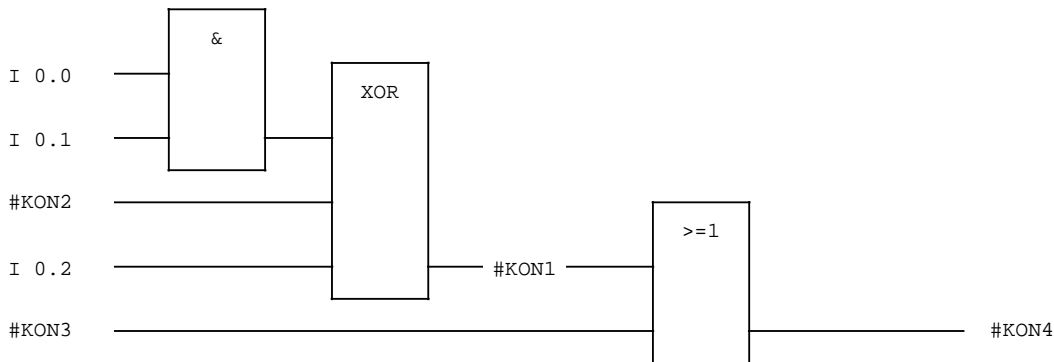


Figure 9-7. Connectors in the IP 265 User Program

Distributors (Insert)

Signal leads which go directly to the output bar may be run to several operands in the output bar (max. 8) via a distributor.



Figure 9-8. Distributor

Connection element (Insert)

Through-connections (e.g. for test purposes) can be generated between operands in the input bar and operands in the output bar without using COM 265 language elements.

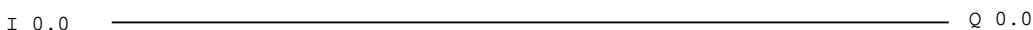


Figure 9-9. Connection Element

9.4 Enter IP 265 User Program

9.4.1 Rules and Recommendations for Programming

In the following sections, you will find

- Information on quantity framework restrictions for the IP 265
- Information on quantity framework restrictions for COM 265
- Tips for optimum utilization of the IP 265
- Rules for referencing operands in the IP 265 user program

Please read each "Note" carefully before attempting to program the IP 265. They can make startup of the IP 265 easier.

Information on quantity framework restrictions for the IP 265

Remember:

The IP 265 user program is stored on the IP 265 in the form of a hardware structure, whereby the language elements are spatially placed on the IP and connected to one another via "connecting lines".

Both the language elements and the connections require memory space (resources) on the IP 265. The resources for the language elements and for the necessary connections are available only to a limited degree.

Note

The IP 265's program capacity is limited. The IP 265 user program's capacity requirements depend on:

- the language elements used **and**
- the interconnection of these language elements in the control system flowchart

COM 265 provides sophisticated aids for estimating the size of the IP 265 user program during the editing phase. An estimation of the IP 265 capacity on-load accorded to the IP 265 user program is displayed on the message line in the "Editor" screen form. The **capacity display** is updated during the entry of COM 265 language elements and operands.

Note

In Section 9.5.1 of the Product Manual you will find an overview table containing the percentage memory requirements for the various language elements, on a number of various interconnection options.

Quantity framework restrictions for COM 265

The COM 265 quantity framework restrictions relating to editing of the IP 265 user program are less "limiting" than those for the IP 265, i.e. COM 265 allows you to edit and test a more complex IP 265 user program than the IP 265 can actually accommodate.

COM 265 must be able to make this possible in order to enable

- off-line testing of two upgraded IP 265s
- additional program structures required exclusively for program testing, i.e. structures which are not to be compiled and loaded into the IP 265.

COM 265 is subject to the following editing restrictions:

Note

- An IP 265 user program may comprise no more than 32 segments.
- A maximum of 31 COM 265 language elements may be interconnected per segment.
- A segment may contain no more than one language element with more than one output (applies to counters). There may be no objects at the right of this language element (remedy: connector).
- In a segment, the outputs of only **one** language element (outer right) can be run to the output bar.
- Identifiers for operands (direct or symbolic addresses) may comprise no more than 24 characters. However, only the first 15 characters are shown in the operand fields of the control system flowchart's margin bars.

Tips for optimum utilization of the IP 265

Always try to solve complex tasks using as little IP 265 program capacity as possible.

Note

- Select practical direct addresses for binary parameters; combine several bit parameters into a byte.
- Initialize timers, counters and comparison functions with constants wherever possible.
Avoid initializing these language elements via parameters.
- If you want to debounce fewer than five 24 V inputs, program in the following address groups:
 - (I 0.y where y=0, 1, 3, 7) or
 - (I 0.y where y=2, 4, 5, 6)

Rules for referencing operands

Note

- Operands are specified with symbolic or direct addresses in the margin bars of the control system flowchart. Inputs and outputs without identifiers ("open" input or output) are identified by the text "????????". Open inputs are treated like constants with a value of "0".
- The direct addresses of the IP 265's inputs and outputs are local, and may therefore also appear in another IP 265's user program and in the CPU's user program and have a different meaning.
- Identifiers (direct or symbolic addresses) for outputs and connectors cannot be "written to" more than once in the IP 265 user program.
- Identifiers for IP 265 outputs are allowed in the output bar only. They cannot not be directly scanned (remedy: distributor/connector).
- The extended inputs/outputs may be referenced as either input or output only (any mix) in the IP 265 user program, i.e. in the control system flowchart, the user may program e.g. either direct address I 2.3 or direct address Q 2.3, but not both (Section 5.1.1).
- Either the fixed addresses of the expansion inputs/outputs or the fixed addresses of the differential inputs may appear in the control system flowchart (Section 5.1.1).

9.4.2 Entries in the Local IP 265 Assignment List

Remember

The operands in the IP 265 user program may have both direct and symbolic addresses (identifiers) in the control system flowchart. Each operand may also be supplemented by an operand commentary.

Note

For your own personal documentation of the IP 265 user program as well as for the initialization of timers, we would recommend programming symbolic identifiers. COM 265 provides an IP 265 assignment list for this purpose which can be invoked in the "Services" form. If you want to program the operands in the IP 265 user program with direct addresses only, and if you do not want to initialize any timers, you can skip this section and proceed directly to Section 9.4.3!

You make up the IP 265 assignment list before you begin entering the control system flowchart for the IP 265 user program.

In the IP 265 assignment list, you assign your symbolic identifiers

- for the actuators and sensors of the process I/O
- for the parameters for data interchange with the CPU and
- for the intermediate values for data interchange with a second IP 265

to the fixed addresses of the IP 265 interfaces.

Later, when you edit your control system flowchart, you enter the symbolic identifiers referenced in the IP 265 assignment list in the "Editor" form's input fields for operands, together with the prefix "-" (or "=" for parameters).

Whenever you move the cursor to a symbolic identifier in the control system flowchart, the IP 265 assignment list line associated with the operand is displayed immediately below the "Editor" form's header. This operand assignment line shows the current operand reference. It also gives you the opportunity to resolve and redefine an existing operand reference without having to exit the "Editor" form (Section 9.4.3).

Note

The IP 265 assignment list is a local operand assignment list, and must not be confused with the global PLC assignment list.

Note

The IP 265 assignment list is a component of the IP 265 user program, and is stored on the **same** file as that program.

The operand assignment list is invoked in the "Services" menu.

"Symbols" screen form

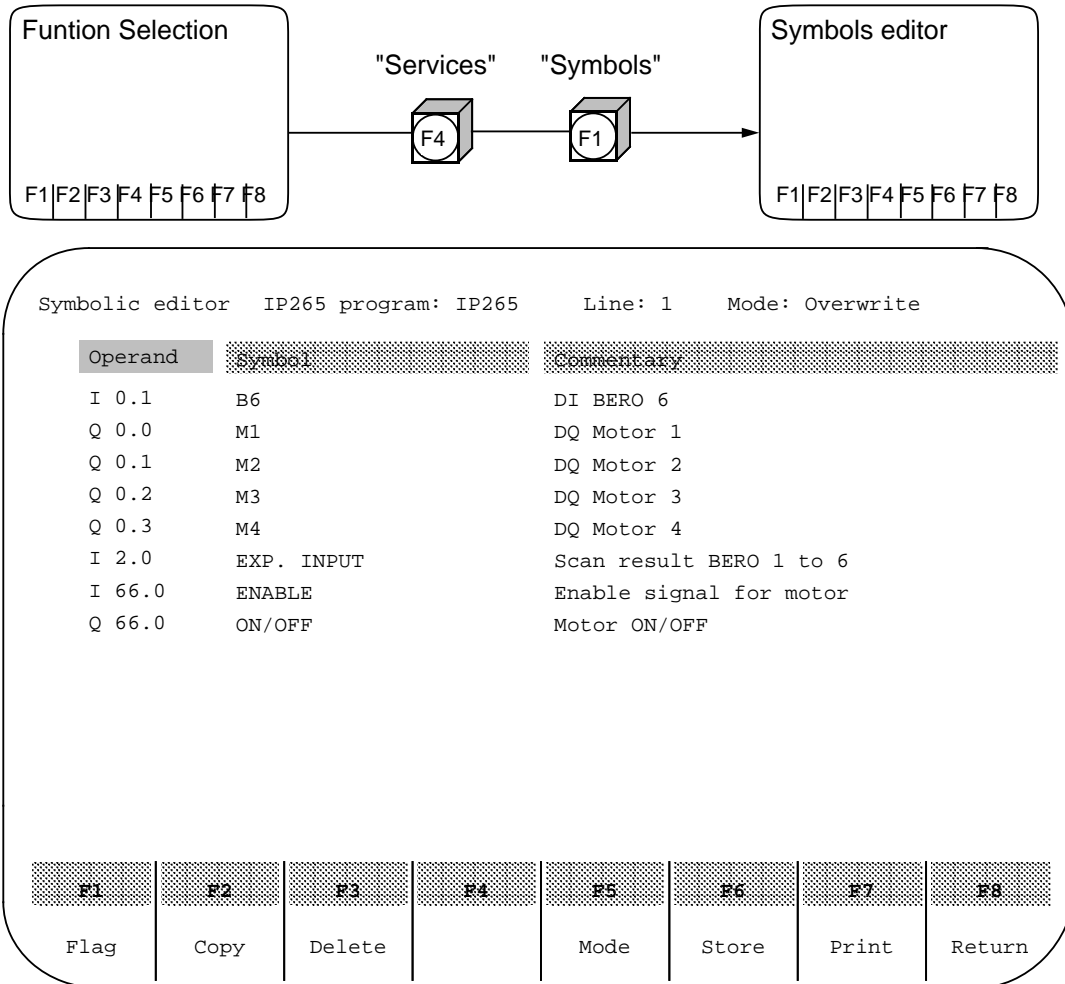
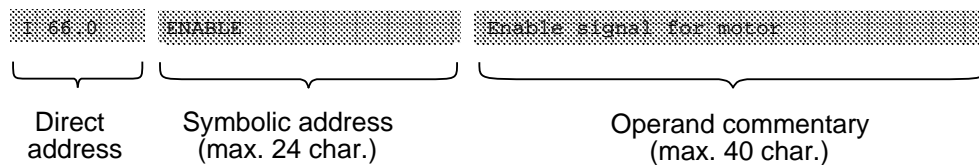


Figure 9-10. "Symbols" Form; Enter Operand Assignment List

Write the operand assignment list. Beginning at the left, enter the identifiers in the table columns of the table:



An operand commentary may be entered in the third column for each operand identifier.

COM 265 supports the input of local operand assignment list entries via function keys. Press <HELP> for information on editing functions and operands.

Note

The "Store" key (<F6>) must be pressed each time modifications are made in the list, also after delete and copy.

The purpose of the IP 265 assignment list is illustrated below, using the initialization of timers as an example.

Example: Initializing timers

Invoke the local operand assignment list (if not already done)

Make some entries in the list. Assign the symbolic identifier of the Preset value to the direct address of an input word parameter, e.g.:

IP 65	TIMER	
-------	-------	--

Press <F6> "Store" to store the operand reference.

Exit the symbols editor with <F8> "Return".

Call the editor in the COM 265 top menu with <F1> "Editor".

Position the cursor in the IP 265 user program to the input field for the timer's Preset value (TV).

Activate the input mode with < 1 >.

Enter the symbolic identifier referenced in the operand assignment list with the prefix "=", then enter the time base (without pressing < 1 > first!), e.g.=TIMER.1

Terminate input with < 1 > or <INSERT>.

Additional information on the IP 265 assignment list:

Note

- If you assigned a symbolic address to an operand in the operand assignment list, COM 265 will always display the operand's symbolic address in the control system flowchart's margin bars.
- A symbolic identifier and operand commentary, once defined, may be overwritten as often as desired in the operand assignment list.
- You can generate individual operand references or resolve and redefine existing operand references in the operand assignment list quickly and easily whenever necessary.
- The operand assignment list can be corrected before, during or after editing of the control system flowchart.
The last entries must be made in the list, at the latest, before compiling the control system flowchart.
- COM 265 automatically stores all addresses specified in the control system flowchart (whether direct or symbolic) in the operand assignment list.
- Direct identifiers used in the control system flowchart cannot be deleted from the operand assignment list.

9.4.3 Entering a Control System Flowchart

In the preceding sections, you have learned something about

- the COM 265 language (Section 9.3)
- the rules and recommendations governing programming (Section 9.4.1) and
- the local IP 265 assignment list (Section 9.4.2)

It is now time to begin entering (editing) the IP 265 user program control system flowchart.

The first point of discussion shall be input of identifiers in the **input fields for operands**. After you have chosen the first COM 265 language element, we shall demonstrate how to:

Enter operand identifiers in the margin bars with automatic cursor control

Enter operand identifiers in the margin bars without automatic cursor control

Enter connectors in the margin bars and logic field

Change symbolic identifiers and operand commentary in the assignment line (line from the local IP 265 assignment list)

Note

One special case in question is the editing of connectors. Connectors may be entered in both the margin bars and the logic field.

All editing functions for the **logic field** in the "Editor" form are listed below:

Insert language element

Insert binary input

Convert language element

Invert binary input/output

Delete language element

Delete input

The control system flowchart editor is invoked in COM 265's "Function Selection" menu.

"Editor" form

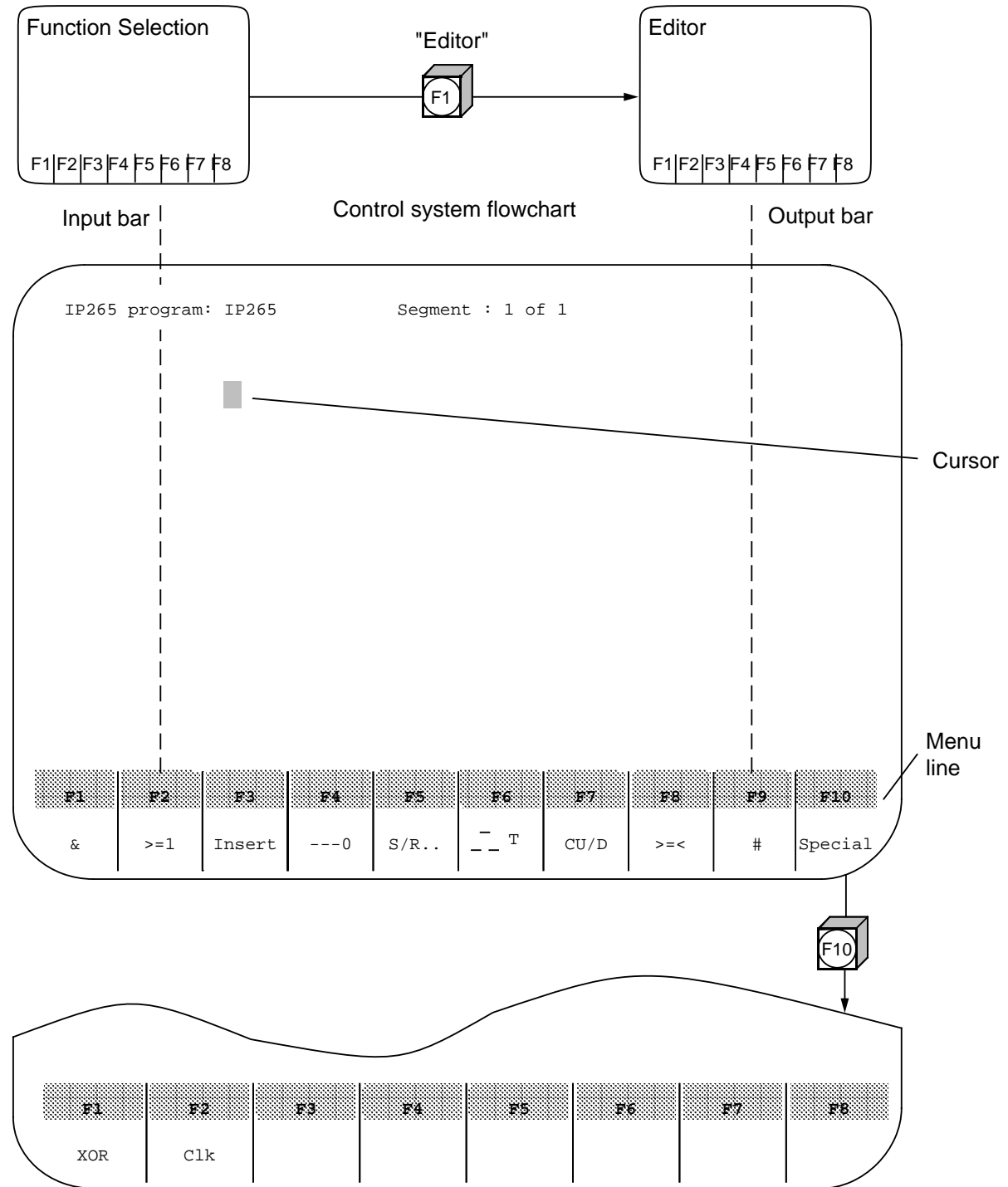


Figure 9-11. "Editor" Form; Enter the Control System Flowchart

The "Editor" form can be exited with <ESC> or <INSERT>.

The following applies to entries in COM 265's "Editor" form:

Before an editing function (e.g. enter operand, insert language element, etc.) can be initiated, you must first select the locations in the control system flowchart at which an action is to take place. To do this, move the cursor to the required location in the input area of the COM 265 screen form.

The following may be selected in the control system flowchart:

- Input fields for operands (including connectors) in the input and output bars
- Connecting lines in the logic field
- COM 265 language elements in the logic field
- Input fields for connectors in the logic field

Selected elements are displayed on the screen in reverse video.

Note

The cursor is moved on a grid-oriented basis. To position the cursor, use the numeric keypad and the <TAB> or <SHIFT> <TAB> key(s) on the programmer keyboard (Section 8.5.4). COM 265 does not support cursor control using a mouse.

In order to acquaint you with the COM 265 control system flowchart editor's many options, we shall first demonstrate all possible operand input options, using a simple language element (an AND operation as an example).

Remember

The following may be edited as operand identifiers in the CSF's margin bars:

- Identifiers for the IP 265's local inputs and outputs
- Identifiers for input and output parameters
- Identifiers for connectors
- Identifiers for constants

With the exception of connectors, operands are located only in the control system flowchart's margin bars. Connectors, on the other hand, may also be located in the logic field between COM 265 language elements.

Note

In Section 9.3.1 you will find an overview of all direct operand identifiers allowed in the control system flowchart's margin bars.

If you want to program symbolic operand identifiers in the control system flowchart's margin bars, you must first enter them in the local IP 265 assignment list (Section 9.4.2).

Entering operand identifiers in the margin bars with automatic cursor control

The operand input fields for each language element edited in the COM 265 "Editor" form are marked with the text "????????".

When a new language element has been selected, the input of its operands is supported by an automatic cursor control facility, i.e. the cursor positions automatically to each operand input field.

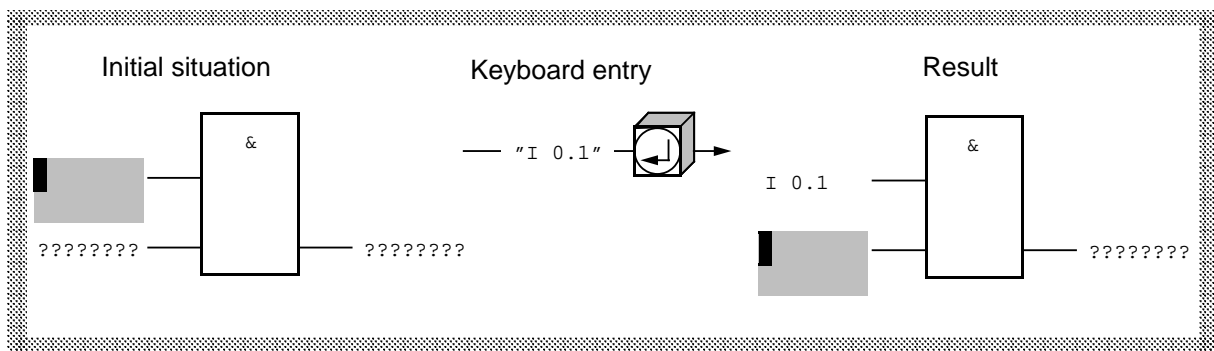
Select the language element. Press the relevant keys in the "Editor" form's function key menu.

As soon as the language element has been selected, the cursor is positioned automatically to the first operand input field. The field comprises two lines, and is displayed in reverse video. The input mode is active. When editing identifiers, proceed as follows:

Enter the direct or symbolic address of the operand on the programmer keyboard.
 Confirm your entry with <1> or <INSERT>.

The cursor automatically goes to the next input field. The text "????????" disappears from the screen, and the input field is displayed as a two-line field and in reverse video. The next operand may now be entered.

Example: Entering operands after selecting "AND" as language element



You can disable automatic cursor control by pressing <ESC>; you can then jump to the next input field with <1>.

Note

If you have assigned symbolic identifiers to the edited operands in the local IP 265 assignment list (Section 9.4.2), COM 265 automatically displays when all direct address entries have been completed. If the cursor is positioned to a symbolic identifier in the marginal bar, the assignment line for that operand in the local IP 265 assignment list is displayed on the monitor screen (Point).

Entering operand identifiers in the margin bars without automatic cursor control

Proceed as follows if automatic cursor control is disabled (<ESC>) and you want to reenter or correct an operand identifier:

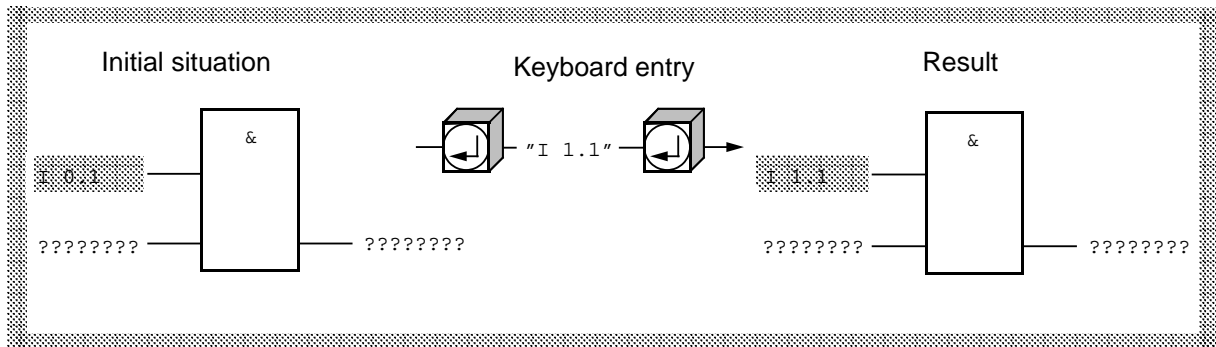
Move the cursor to the input field

Activate Input mode by pressing <1>

Enter the direct or symbolic address of the operand on the programmer keyboard

Confirm your entry with <1> or <INSERT>. (Or you can revoke the entry by pressing <ESC>).

Example: Converting identifier "I 0.1" into "I 1.1"



Note

Before entering or converting a direct or symbolic address, you must always activate the Input mode for the editing field by pressing <1>. In Input mode, the input field appears as a two-line field in reverse video.

COM 265 always checks your entries. If you have entered an invalid identifier, an appropriate error message is displayed on the "Editor" form's message line.

Input fields without identifiers ("open inputs") continue to show "????????". Open inputs in the IP 265 user program are treated like constants with a "0" value.

Note

An operand, once entered, can be overwritten as often as required. However, it is not possible to convert identifiers for differential inputs into those for expansion inputs or vice versa. Examples:

- Valid conversion: I 0.0 (24 V input) I 2.1 (24 V input)
- Valid conversion: I 0.0 (24 V input) I 71.1 (input parameter)
- Invalid conversion: I 1.1 (diff. input) I 2.6 (expansion parameter)

Entering connectors in the margin bars and logic field

Connectors may be entered in both the margin bars and the logic field of a control system flowchart.

Entering connectors in the margin bars:

Input activities depend on whether automatic cursor control is enabled or disabled.

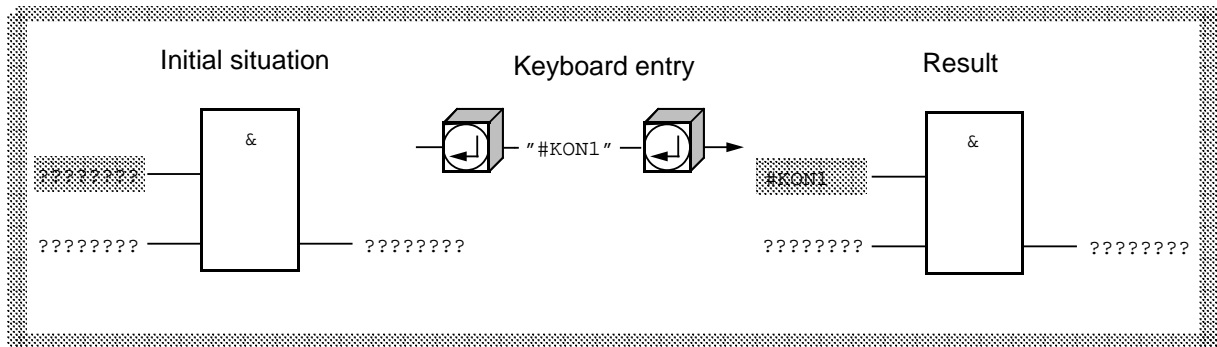
Automatic cursor control is enabled immediately following selection of the language element. When you have reached the input field for the connector, edit the connector identifier as follows:

Enter the "#" character (programmer keyboard) and follow it with a symbolic text.
 Confirm your entry with <1> or <INSERT>.

If automatic cursor control is disabled and you want to convert an identifier into a connector or correct an identifier, proceed as follows:

Move the cursor to the input field
 Activate the Input mode with <1>
 Enter the "#" character (programmer keyboard) and follow it with a symbolic text
 Confirm your entry with <1> or < INSERT>

Example: Entering connector "#KON1' in a marginal bar



The identifier for connectors ("#" plug "symbolic string") may comprise no more than 25 characters. If you edit an identifier consisting of more than 16 characters, "*" is displayed as the 16th character when the entry is confirmed.

Entering connectors in the logic field:

Move the cursor to the connecting line in which the connector is to be inserted (input or output of a language element).

Select the "#" element in the softkey menu by pressing <F9>. The input field, with the "#" character, is displayed at the required position.

Add a symbolic identifier.

Confirm your entry with <1> or <INSERT>.

Example: Connectors in the input bar and logic field

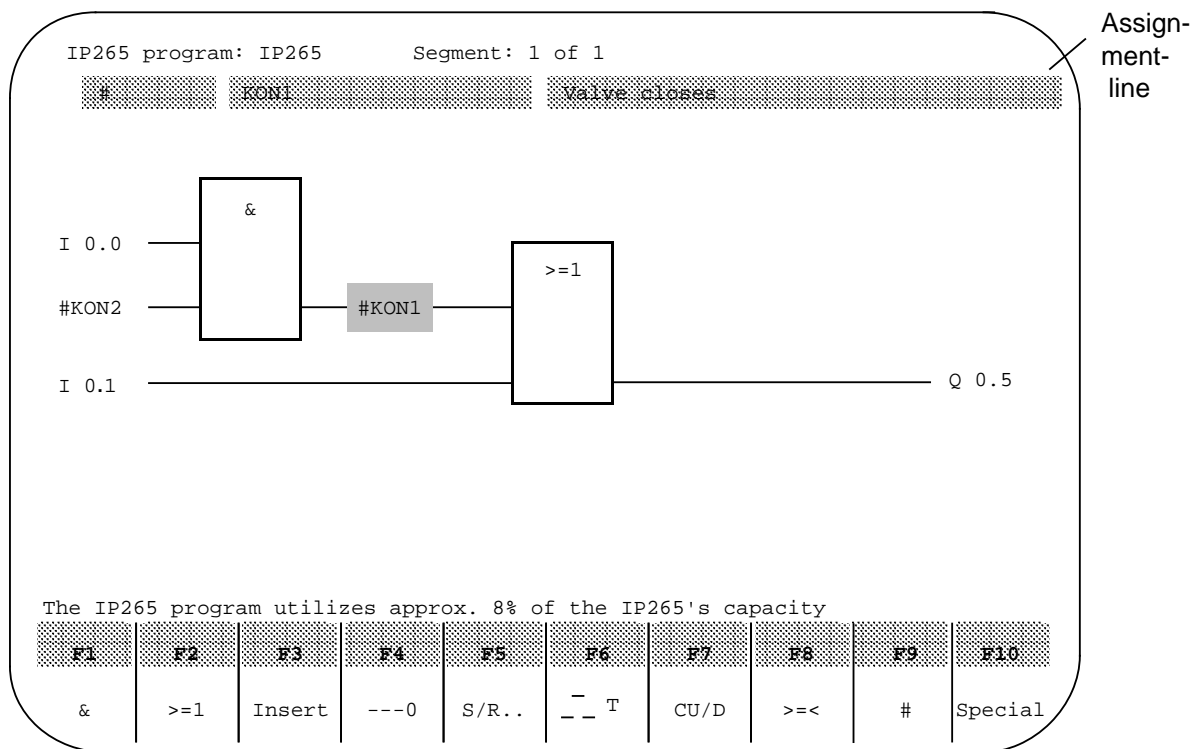


Figure 9-12. "Editor" Form; Enter Connectors

Note

Connectors can be "written" only once in the IP 265 user program, but they can be "scanned" as often as required and at various points in the program.

Entering operand commentary for connectors:

Immediately upon completing a connector entry, and whenever the cursor is positioned to a connector in a control system flowchart, an assignment line is displayed beneath the header (Point).

If you press the <CORR> key in Input mode, the cursor jumps from the current input field to the assignment line.

This allows you to enter a commentary (max. 40 characters for the connector symbol ("KON1"), e.g. "Valve closes".

You can confirm your entry or modify the commentary with <1> or <INSERT>, or exit the assignment line without changing the original contents with <ESC>.

Modifying symbolic identifiers and operand commentary on the assignment line

Whenever you move the cursor to a symbolic identifier in a control system flowchart, the line from the IP 265 assignment list for that operand is displayed immediately below the "Editor" form's header.

"Editor" form

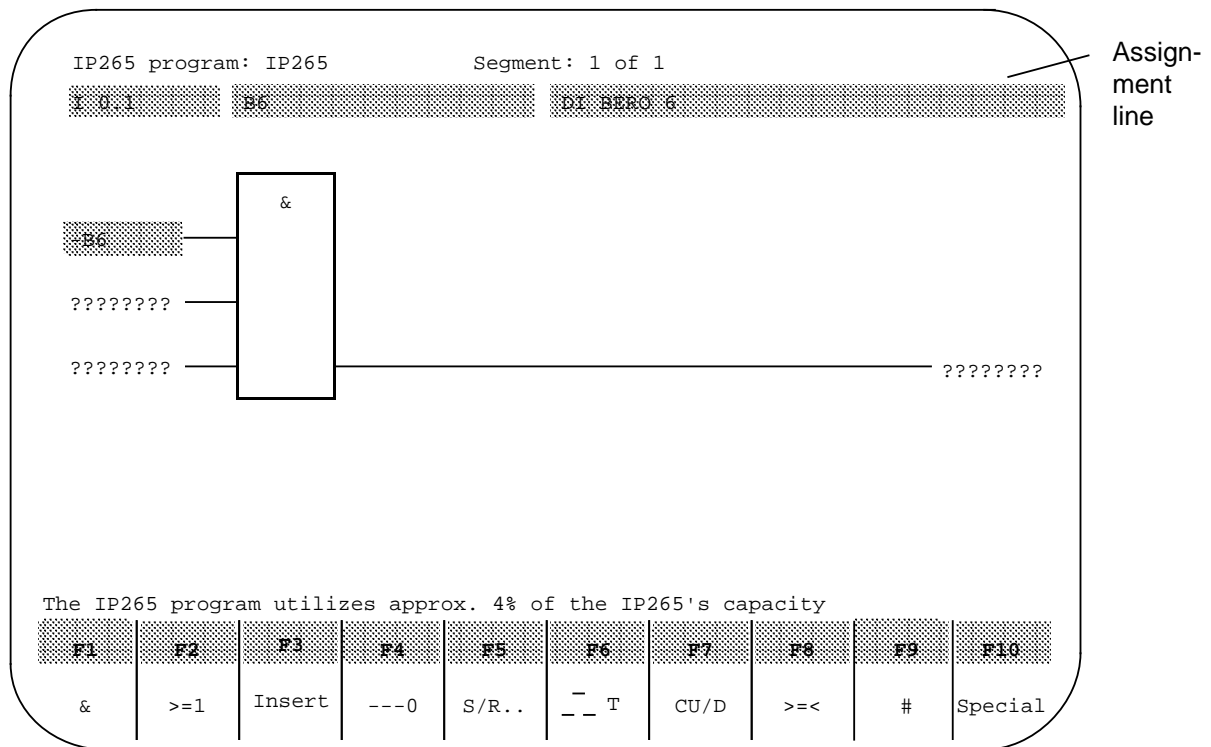


Figure 9-13. "Editor" Form; Enter Assignment Line

In the operand's assignment line, you can:

- view the assignment of direct to symbolic address made in the IP 265 assignment list (Section 9.4.2)
- dissolve the existing assignment of direct to symbolic address
- generate a new assignment of direct to symbolic address
- modify the existing operand commentary
- generate a new operand commentary

The same input options are available as for editing of the IP 265 assignment list (Section 9.4.2). Proceed as follows to make entries in the assignment line:

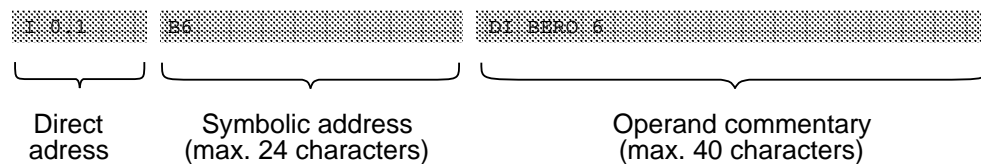
Move the cursor to the input field

Activate the Input mode with <1>

Press <CORR>.

The cursor jumps to the assignment line for the current input field.

Type the assignment line (you can jump to the next (previous) input field with <TAB> (<SHIFT><TAB>)):



To continue:

Confirm entries in the assignment line with <1>. The current identifier is transferred to the operand input field in the marginal bar.

Exit the assignment line without changing the original contents of the input field with <ESC>.

Terminate input with <1> or <INSERT>.

In addition to the symbolic COM 265 identifiers, the following symbols are automatically faded into the input fields for operands in the margin bars:

Symbol	"_"	"="
Prefix for	symbolic identifier for an IP 265 input/output	symbolic or direct identifier for a parameter

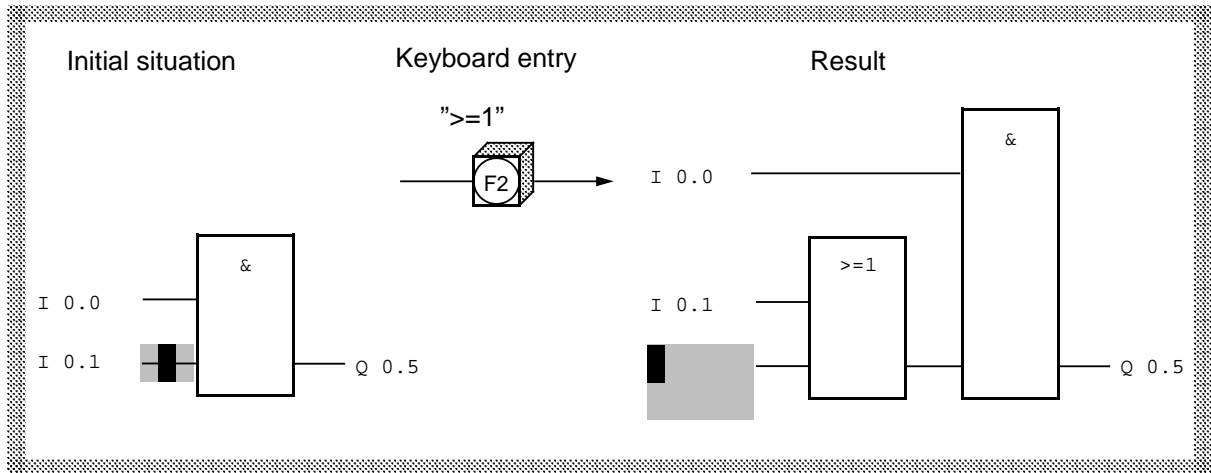
The input fields in the margin bars can accommodate 16 characters. If you enter a symbolic identifier which exceeds 16 characters, the 16th character appears as "*" in the operand input field when the entry is confirmed.

Inserting a language element

Move the cursor to the connecting line in which the language element is to be inserted (omit in the case of an empty segment).

Select the language element to be inserted from the choices given on the menu line and press the associated function key.

Example:



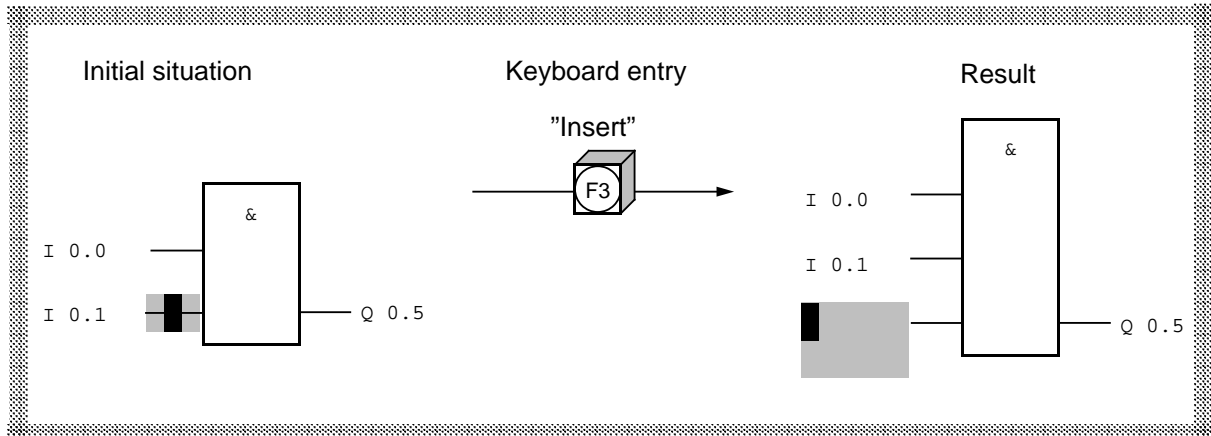
Inserting a binary input

You may insert additional inputs (as many as 8) for language elements AND, OR and XOR.

Move the cursor to an input of the language element for which an additional input is to be generated.

Select the "Insert" function in the function key menu by pressing function key <F3>.

Example:



Converting a language element

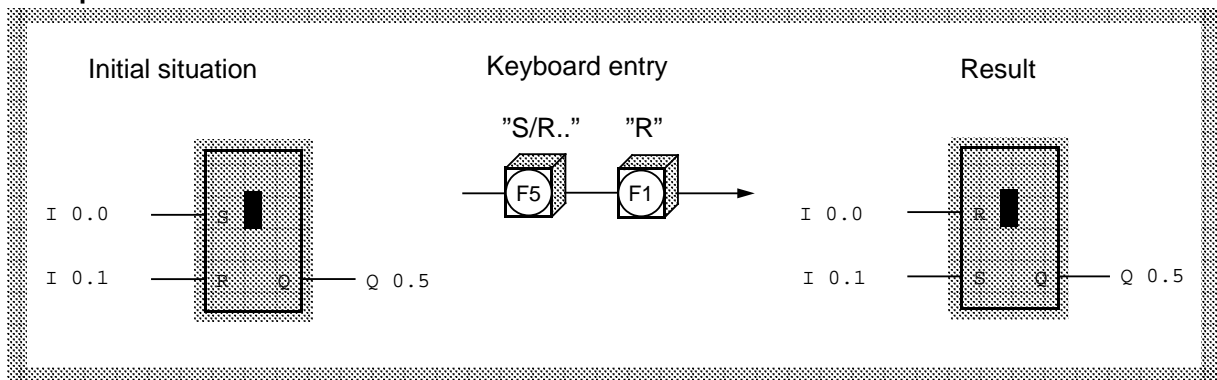
You can convert a language element in a control system flowchart into another, type-conforming language element. Language elements are type-conforming when the number and data type of their inputs and outputs are identical.

A Comparison operation can be converted into another Comparison operation (for instance, ">" into "<"). By the same token, a timer operation, a binary logic operation or a set/reset operation can be converted into another operation of the same type.

Move the cursor to the language element to be converted. This language element is then displayed in reverse video.

Select the new, type-conforming language element from the choices given on the menu line by pressing the associated function key.

Example:

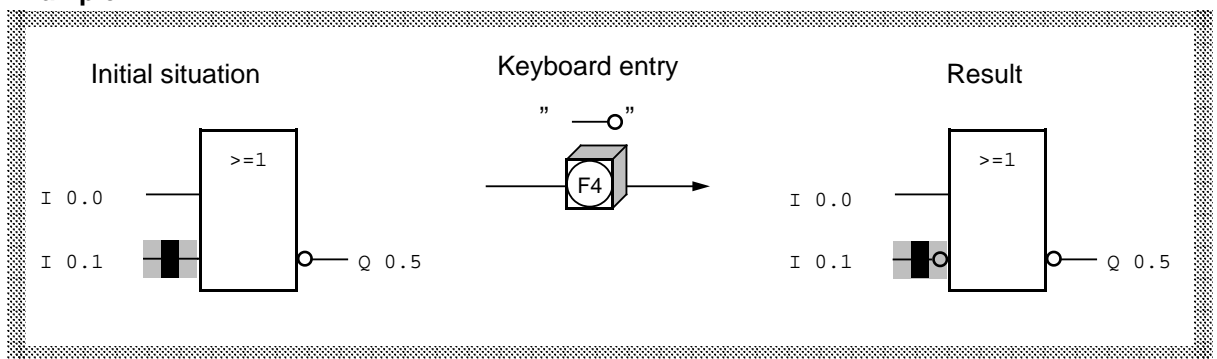


Inverting a binary input/output

Move the cursor to the connecting line.

Selected the "Invert" function by pressing function key <F4>.

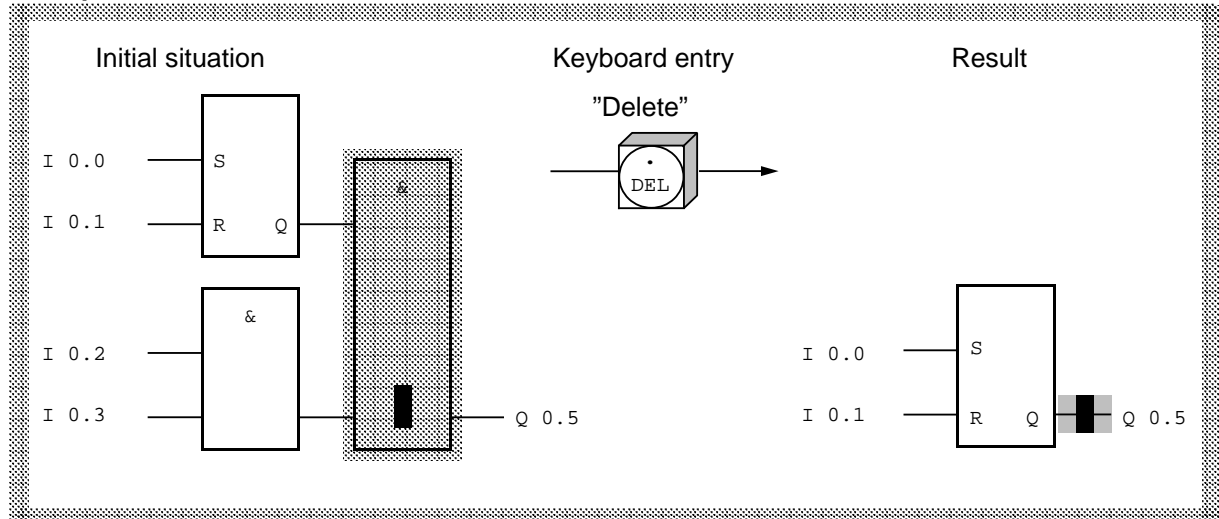
Example:



Deleting a language element

Move the cursor to the language element to be deleted.
 Press the key on the programmer keyboard's numeric keypad.

Example:



Note

Language elements on the "open" inputs and outputs of the language element to be deleted will also be deleted.
 In the event of a reassignment of connections with information pertaining to different data types, the segment portion at the point of conflict will also be deleted.

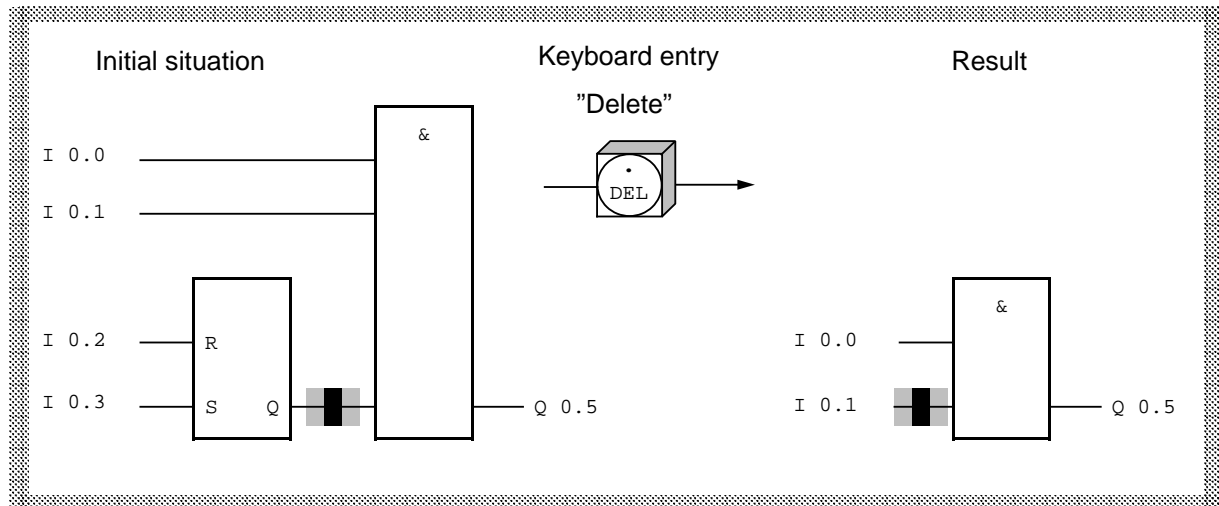
Deleting an input

Additionally inserted inputs (>2 inputs) for expandable language elements AND, OR and XOR can be removed through deletion.

Set the cursor to the connecting line to be deleted.

Press on the programmer keyboard's numeric keypad.

Example:



Note

Language elements at the left of the input to be deleted will also be deleted.

9.4.4 Entering the Segment Name and Segment Commentary

In addition to the program logic, you can also input the following in order to document the IP 265 user program:

- A header for each segment (segment name or segment header)
- A commentary for each segment in the IP 265 user program (segment comments or segment commentary)

Note

Segment header and segment comments are components of the IP 265 user program, and are stored on the **same** file.

Segment name (segment header)

Proceed as follows to enter a segment name:

Press the **<COM>** key **once** only.

The input field for the segment name is displayed in reverse video on the "Editor" form's header line (Figure 9-14).

Type the segment name on the programmer keyboard.

Confirm your entry with **<1>**.

Press **<ESC>** to exit the input field without changing its original contents.

The segment name may comprise up to 24 characters.

Segment commentary

Proceed as follows to enter a segment commentary:

Press **<COM>** **twice** in succession in the "Editor" form.

The "Commentary" input form is displayed on the monitor.

Type your commentary text on the programmer keyboard.

The "Commentary" form is invoked in the editor:

"Commentary" form

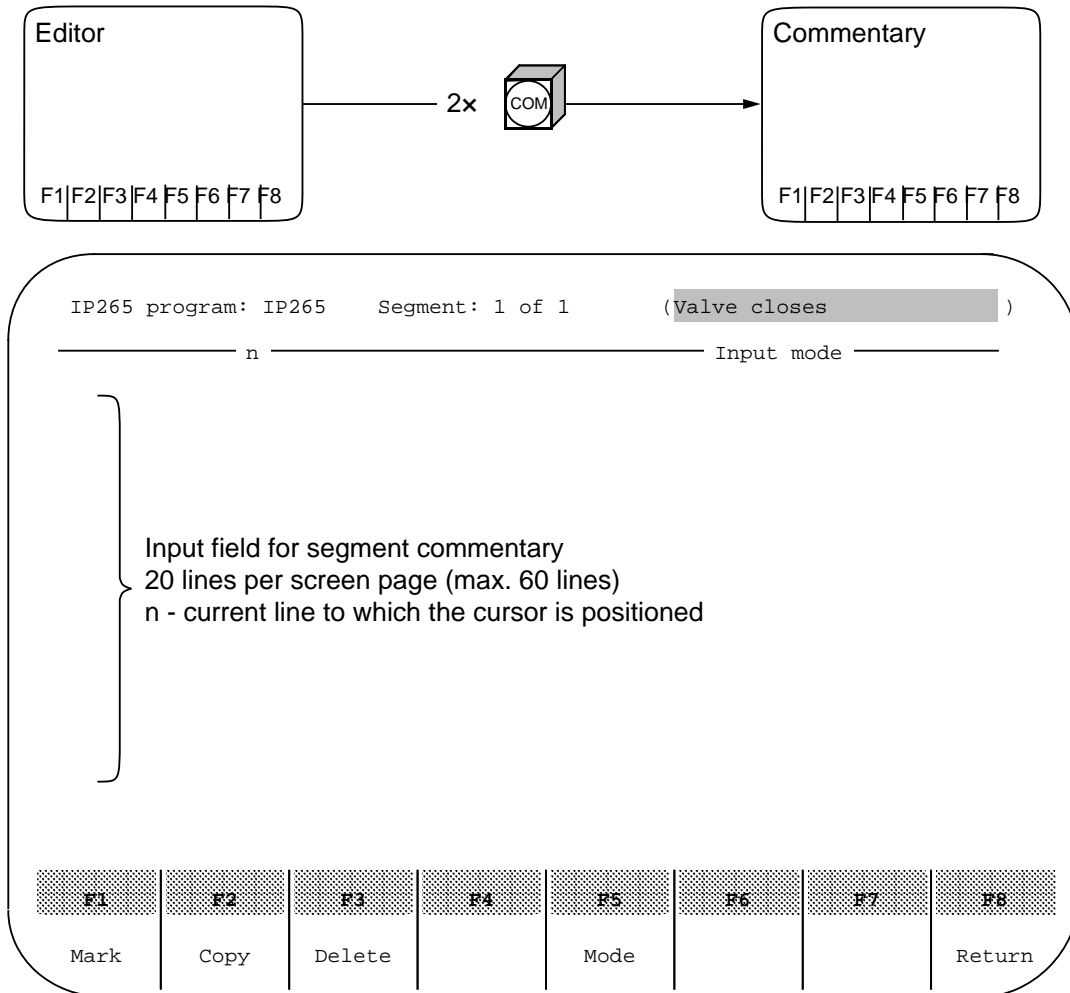


Figure 9-14. "Commentary" Form; Enter Segment Commentary

COM 265 supports entry of the segment commentary via function keys. Press <HELP> for information on the editing functions.

Further procedures:

Terminate input with <F8>.

Press <1> twice in succession to confirm entry of the segment commentary. Press <ESC> twice in succession to exit the "Commentary" form without changing the original contents.

9.5 Compiling the IP 265 User Program

Note

The IP 265 user program must be compiled before it can be stored on a memory sub-module or loaded into the IP 265 over the I/O bus.

Prerequisites

- The IP 265 user program to be compiled must be specified in the "Defaults" form.

Initiating a compiler run

Select the "Compile" function in the "Function Selection" menu by pressing <F2>.

"Compile" form

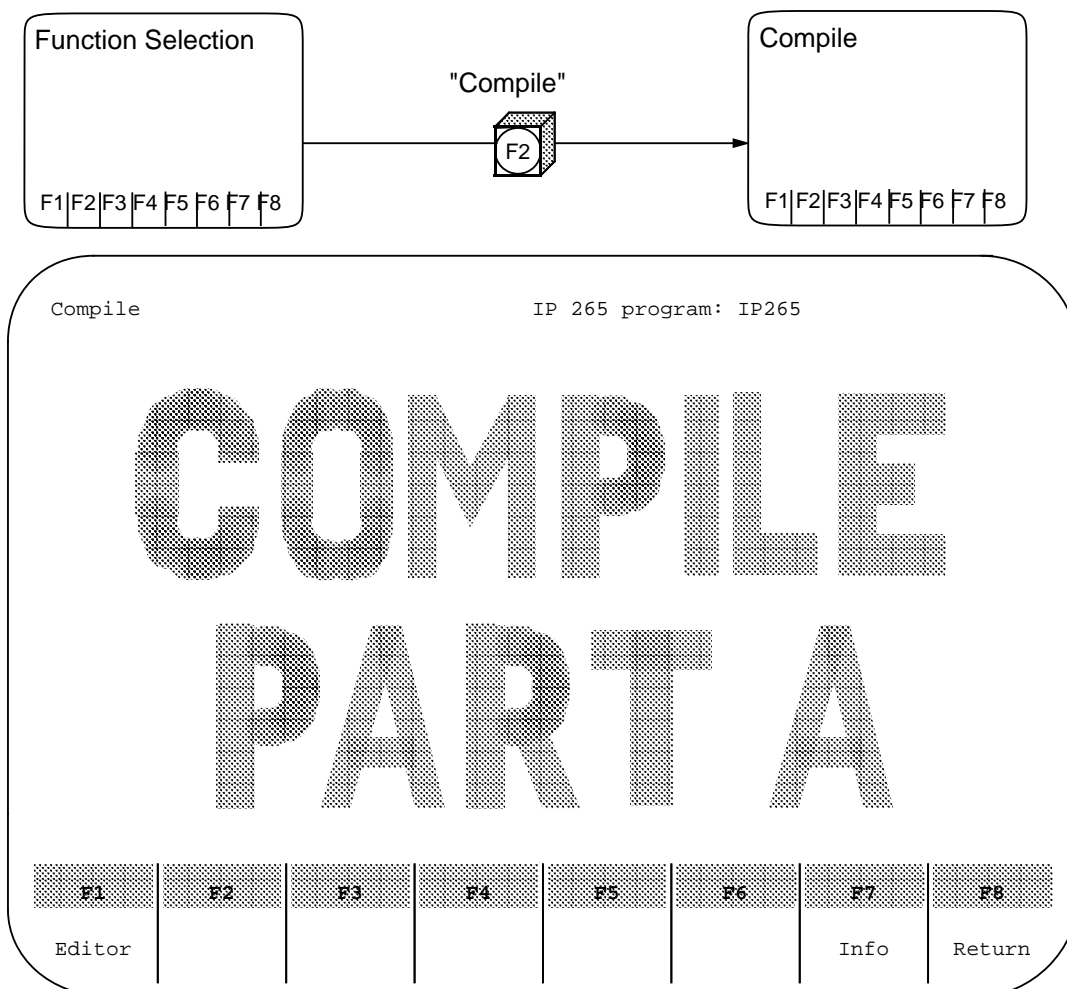


Figure 9-15. "Compile" Form; Placing COM 265 Language Elements

Once the compilation run has been started, COM 265 handles the run autonomously. A compilation takes place in three phases:

- Phase A: Placing the COM 265 language elements
- Phase B: Interconnecting the COM 265 language elements
- Phase C: The resulting IP 265 user program is translated into data relevant to the IP 265

The phase currently executing is displayed on the monitor.

Errors

If the IP 265 user program cannot be compiled, COM 265 aborts the compilation prematurely and displays an error message on the monitor screen. You can press <F7> "Info" to screen a Help window which provides information on the cause of error and recommendations on how it can be rectified.

Some examples of error messages displayed by COM 265:

- "The IP265 program is too long!"
- "Illegal connection of the comparator in segment "3""
- "The IP265 program has not yet been generated (blank program)"

To allow quick corrections, you can go straight from the compiler to the editor (Section 9.6.2).

Result message

If the IP 265 user program was free of errors and not too big, the following message is displayed when the compilation has been successfully completed:

"Program has been compiled without error, and utilizes xx% of the IP's capacity."

The value 'xx%' shows how much of the IP 265's capacity is taken up by the IP 265 user program.

9.5.1 IP 265 User Program Load on the IP 265

Remember

The IP 265's program capacity is limited. The IP 265 user program load on the IP 265 is dependent on two factors:

- The amount of memory needed for the COM 265 language elements in the IP 265 user program
- The interconnection of the COM 265 language in the IP 265 user program

Table 9-13 provides an overview of the load (in %) placed on the IP 265 by the various COM 265 language elements and their interconnection. This table will help you estimate whether your IP 265 user program can be compiled, or whether it is too big for the amount of space available in the IP. The capacity load display in COM 265's "Editor" form is based on this table.

Table 9-13. IP 265 Load Effected by COM 265 Language Elements and their Interconnection

COM 265 language element	Type of interconnection						IP 265 load *	
Arbitrary binary logic operation with 5 inputs							4 %	
Arbitrary binary logic operation with 6 to 8 inputs							8 %	
Set/reset operations: - RS flipflop - Delay - Binary scaler - Edge flag - Clock-pulse generator **							4 %	
	Time value as constant						12 %	
Timer operations **	Time value as input parameter						24 % (typ. SD: 28 %)	
Counter operations	Connected as		Preset as		Count connected to			
	CU	CD	Con- stant	Input param.	Comparator or not	Output param.		
	X	X	X		X			20 %
	X		X		X			20 %
		X	X		X			16 %
	X	X		X	X			32 %
	X			X	X			28 %
		X		X	X			28 %
	X	X	X			X		28 %
	X		X			X		28 %
		X	X			X		24 %
	X	X		X		X		36 %
	X			X		X		36 %
	X		X		X	32 %		
Comparison operations	Comparison of BI output with							
	Constant			Input param.				
	=/			X				4 %
	X			X				16 %
> <	X			X			8 %	
				X			20 %	
Debouncing of I 0.0/I 0.1/I 0.3/ I 0.7 (individually or in combination)							8 %	
Debouncing of I 0.2/I 0.4/I 0.5/ I 0.6 (individually or in combination)							8 %	
Each param. byte in the output data (max. 8 bin. input params)							12 %	
Each parameter byte in the input data (max. 8 binary output params)							8 %	

* Relating to the program capacity of the IP 265: 100 %

** Base load at 0.5 / 2 / 16 / 64 kHz: + 8 %
Base load at 1 / 10 / 100 Hz: + 12 %

9.5.2 What if there is an IP 265 Overload?

If COM 265 aborts the compiler run because the IP 265 user program is too big, you can assess the best measures for compressing the program by adding up the percent values shown in Table 9-13.

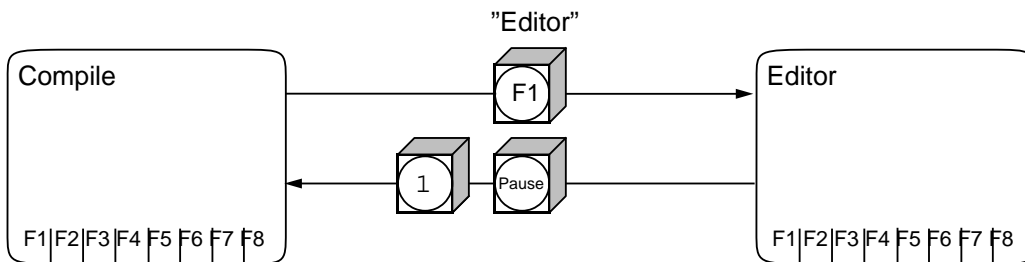
The following measures are recommended when the 100 % load limit has been exceeded:

- If parameters have been used in the IP 265 user program, check the following:
 - Were the direct addresses of binary parameters chosen with care? Groups of bit parameters should be combined into a byte!
 - Can the parameter initialization of timer/counter or Compare operations be replaced by the use of constants?
- If 24 V inputs have been configured for debouncing in the IP 265 user program, check to see whether the direct addresses of the debounced inputs can be made part of one of the following address groups:
 - I 0.y with y=0, 1, 3, 7 or
 - I 0.y with y=2, 4, 5, 6
- Consider whether some parts of the IP 265 user program could be optimized to put less load on the IP 265 (Section 13.2, "Programming Aids").

Other measures:

- Switching back and forth from "Compiler" to "Editor"
The compiler level has no provisions for editing functions such as changing an operand identifier or deleting a language element, but it is still possible to make any necessary changes quickly and easily:

You can switch back and forth between the Compiler and the Editor.



Confirm any changes in the "old" COM 265 screen form with <1> (see above) or revoke them with <ESC>.

10 Testing and Simulation with COM 265

10.1	Off-Line Simulation of the IP 265 User Program	10- 1
10.1.1	General Information on Simulation with COM 265	10- 1
10.1.2	Starting the Simulator and Simple Simulation of a COM 265 Language Element	10- 4
10.1.3	Generating Simulator Settings	10- 6
10.1.4	Resetting Simulator Settings	10- 14
10.1.5	Symbols for Simulator Settings	10- 15
10.1.6	Clocked Simulator Control	10- 17
10.1.7	Text Representation	10- 18
10.2	On-Line Testing of the IP 265	10- 20
10.3	Wiring Test	10 -23

Figures		
10-1	"Simulator" Form; CSF Format	10- 4
10-2	"Simulator" Form; Text Format	10- 19
10-3	"Online" Form	10- 21
Tabellen		
10-1	Simulating a Reset Dominant Flipflop	10- 5
10-2	Triggering a Signal State Change	10- 12
10-3	Resetting Simulator Settings	10- 14
10-4	Symbols for Simulator Settings	10- 15
10-5	Displaying Signal and Value Manipulations	10- 16

10 Testing and Simulation with COM 265

COM 265 provides many possibilities for testing the IP 265 user program and the execution of the user program in the IP 265.

The COM 265 **simulator** makes it possible to simulate most of the interconnections in the IP 265 user program off-line, i.e. without using a PLC or IP 265 (Section 10.1).

In a second step, the logically correct user program can be **tested on-line** under real-time conditions (conditions governing subgroup control within the plant as a whole) in the PLC and with process signals (Section 10.2).

Independently of the IP 265 user program, you can also carry out a **wiring test** to check the interfaces to the process (Section 10.3).

10.1 Off-Line Simulation of the IP 265 User Program

Before loading it into the IP 265, the IP 265 user program should be tested off-line on the programmer. This makes it possible to detect programming errors quickly without using process I/Os.

10.1.1 General Information on Simulation with COM 265

COM 265 provides a special simulator which simulates the IP 265 in the programmer.

What can you do with the COM 265 simulator?

- Controller simulation
You can specify signal states for simulating the control system flowchart. The results of the simulation are the control system flowchart's new signal states, which are displayed on the programmer screen.

You can specify signal states "0" and "1" for

- IP 265 inputs
- binary input parameters and
- internal connections between COM 265 language elements.

You can also manipulate actual values for latching COM 265 language elements without having to exit the simulator.

- Simple form of process simulation
You can loop IP 265 outputs in the control system flowchart back to IP 265 inputs. When this is done, the output signal is looped back after a (configurable) delay, making it possible to simulate e.g. limit switches.

The simulator's dynamic response

The COM 265 simulator simulates the actual real-time conditions of the IP 265 in the programmer, i.e. the interconnections in the IP 265 user program are clocked at 128 kHz (IP 265 clock pulse: $1/128 \text{ kHz} = 7.8 \text{ fs}$). In order to be able to visually monitor signal state changes, there are two ways of defining the interval between two clock pulses on the programmer monitor itself:

- You can specify how great the time difference between two clock pulses should be (configurable), and the simulator then automatically clocks the connection.
- You can clock the connection manually in single-step mode (default setting).

Note

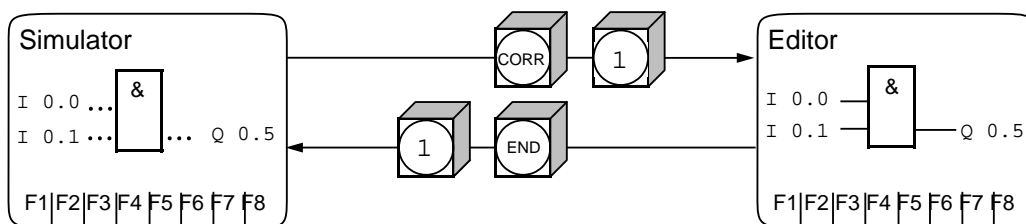
When you change the signal state of a binary variable, the change does not affect the control system flowchart until the next IP 265 clock pulse.

The COM 265 simulator simulates the IP 265's "**internal**" program scan. Delays such as input and output delays (Section 4.2) and configured debouncing times are not taken into account.

Additional performance characteristics of the simulator

- Switching from "Simulator" to "Editor"
The simulator level has no provisions for editing functions such as changing an operand identifier or deleting a language element, etc. However, minor modifications, such as those required while testing, are nonetheless possible, and can be carried out quickly and easily:

You can go from the simulator directly to the COM 265 editor.



During such a change, changes in the "old" COM 265 screen form can be confirmed with <1> (see above) or revoked with <ESC>.

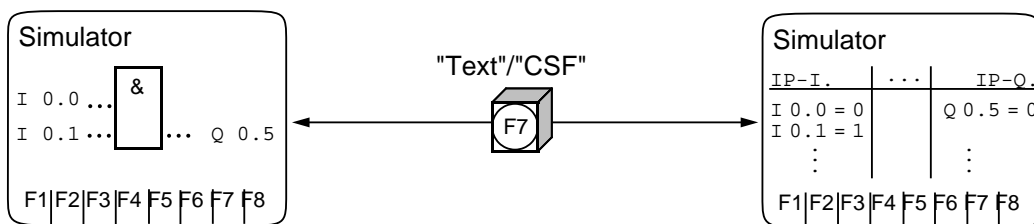
Switching from the COM 265 "Simulator" to the COM 265 "Editor" is context-dependent, i.e. the current section of the control system flowchart remains on screen; only the header and the function key menu change (also, you can always go directly to the editor while in the simulator's Text format).

- CSF format and text format during simulation
You can choose between two formats when simulating the IP 265 user program:
 - Control system flowchart format (default)
The control system flowchart appears as in COM 265's "Editor form"
 - Text format
IP 265 inputs, IP 265 outputs, parameters and selected COM 265 language elements are displayed, together with their current signal states and values, in table form.

When the simulator is started, the control system flowchart format is automatically enabled. You can change to text format and back as follows:

CSF format (default)

Text format



All information presented in the following sections is based on CSF format. You will find more detailed information on text format in Section 10.1.7.

- Symbols during simulation
The simulator supports you with a large number of symbols. In Section 10.1.5 ("Symbols for Simulator Settings") you will find an overview of all symbols appearing on the programmer screen during the on-line test.
- Printing out the IP 265 user program in the simulator level
It is possible to print out the on-screen segment in CSF format. To do so, press <CTRL> <P>. In text format, you can print out the entire IP 265 user program by pressing the same two keys.

10.1.2 Starting the Simulator and Simple Simulation of a COM 265 Language Element

Every IP 265 user program can be simulated off-line without any restrictions whatsoever.

Note

Even IP 265 user programs which cannot be compiled because they are "too big" (IP 265 load > 100 %) can be simulated. You must subsequently subdivide the IP 265 user program between two IP 265s.

Prerequisites for the off-line test

The IP 265 user program to be simulated off-line must be selected in COM 265's "Defaults" form.

"Simulator" form

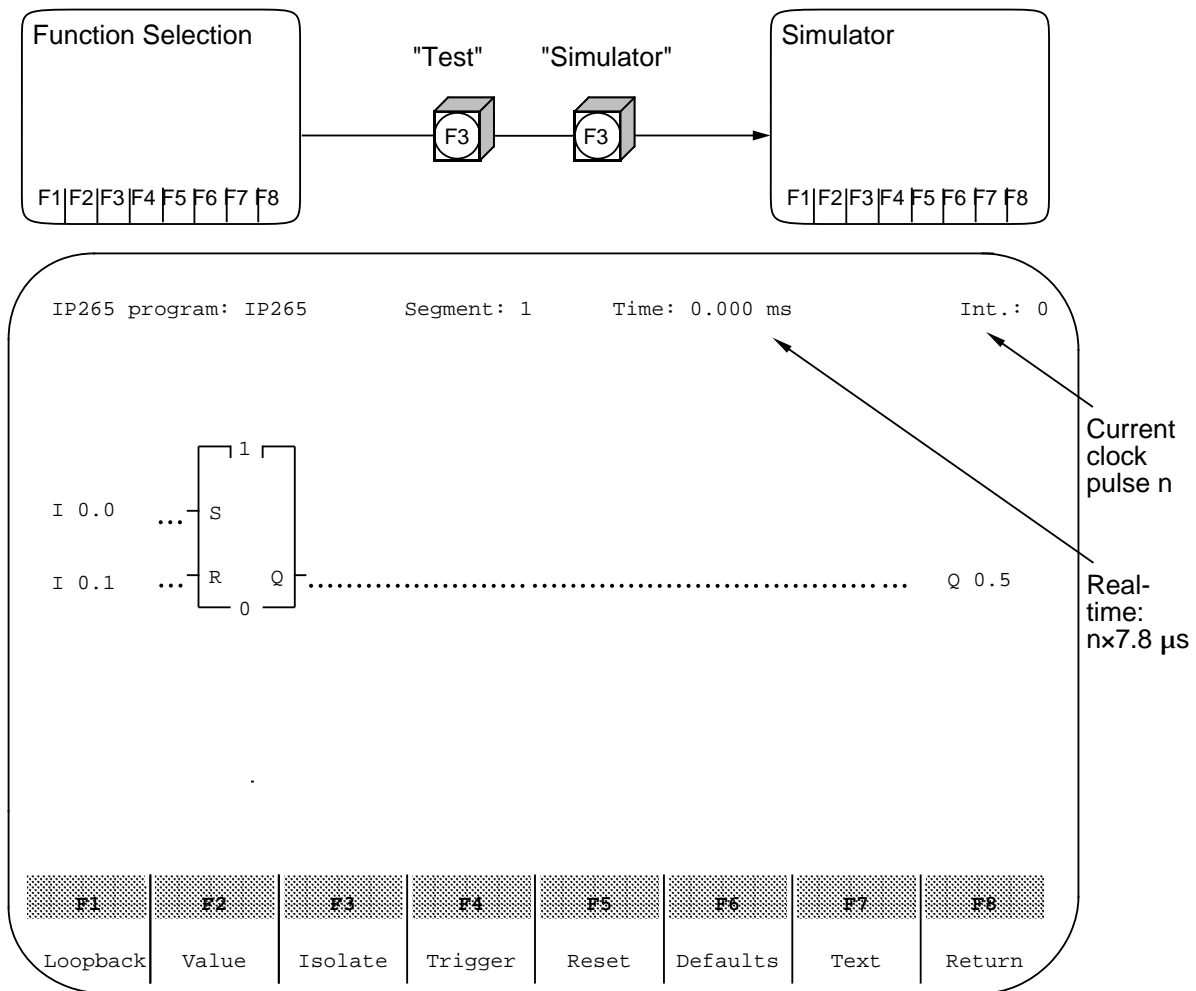


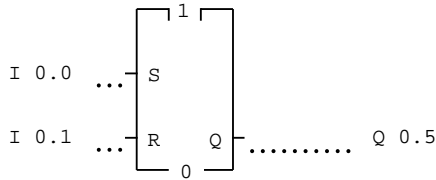
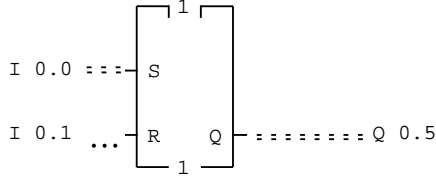
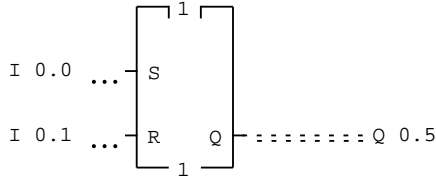
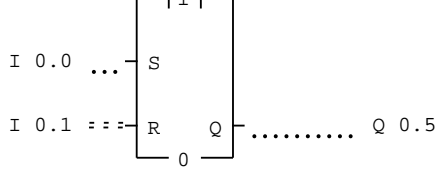
Figure 10-1. "Simulator" Form; CSF Format

When invoked, the simulator is set for control system flowchart format. All information presented in the following sections is based on this format.

To acquaint you with the COM 265 simulator, the example below deals with the simplest form of simulation. In subsequent sections, you will find detailed information on all of the many possible simulator settings. You yourself must decide which simulator settings are best for your program test.

Example: Simulating a Reset Dominant Flipflop

Table 10-1. Simulating a Reset Dominant Flipflop

User actions	Display
<p>In order to simulate a flipflop, you must carry out two activities. You must:</p> <ul style="list-style-type: none"> Specify the signal states of the Set input and the Reset input Clock the Set/Reset operation manually 	<p>Initial situation</p> 
<p>a) Setting an RS flipflop</p> <p>You can change the signal state at Set input I 0.0 from "0" to "1" by entering <0>+<0>*. Press the space bar to generate an IP 265 clock pulse. The signal state change is injected into the circuit. Press the space bar to generate a second IP 265 clock pulse.</p> <p><i>The RS flipflop has been set.</i></p>	<p>RS flipflop set</p> 
<p>b) Resetting a Set input</p> <p>You can change the signal state at Set input I 0.0 from "1" to "0" by entering <0>+<0>*. Generate two IP 265 clock pulses by pressing the space bar twice in succession.</p> <p><i>The RS flipflop remains set.</i></p>	<p>Latching RS flipflop</p> 
<p>c) Resetting an RS flipflop</p> <p>You can change the signal state at Reset input I 0.1 from "0" to "1" by entering <0>+<1>*. Generate two IP 265 clock pulses by pressing the space bar twice in succession.</p> <p><i>The RS flipflop is reset.</i></p>	<p>RS flipflop reset</p> 
<p>You can reinstate the initial situation by changing the signal state at Reset input I 0.1 from "1" to "0" and pressing the space bar.</p>	

* Table 10-2

10.1.3 Generating Simulator Settings

Prior to off-line simulation of the IP 265 user program, you must specify the simulator settings. COM 265 simulator options include:

- Looping IP 265 outputs back to IP 265 inputs
- Isolating part of the control system flowchart
- Specifying actual values via function keys
- Specifying signal states
- Triggering timer and counter functions

Note

Simulator settings do not immediately affect the control system flowchart. The new signal states and actual values do not take effect until the next IP 265 clock pulse (Section 10.1.6).

Looping IP 265 outputs back to IP 265 inputs (<F1> "Loopback")

An IP 265 output can be looped back with a delay to an IP 265 input, e.g. to simulate the function of a limit switch.

Two delay times, each of which may differ from the other, must be specified for each loopback:

- Delay time for a rising edge at the IP 265 output (signal state change from "0" to "1")
- Delay time for a falling edge at the IP 265 output (signal state change from "1" to "0")

The simulator then loops the edge change back to the IP 265 input after the programmed delay time has expired.

Note

A looped IP 265 input can be affected interactively via function key <F4> in the "Simulator" form's softkey menu ().

Example: Looping output 0.5 back to input 0.0

Move the cursor to input 0.0 in the control system flowchart's input bar.

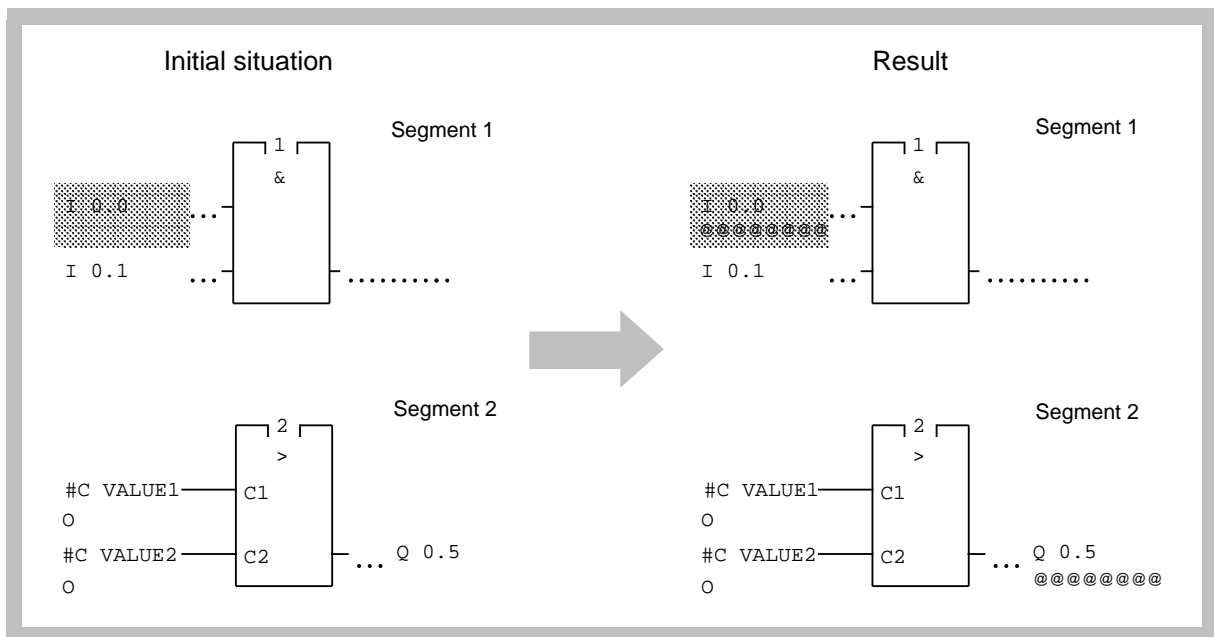
Select "Loopback" from the softkey menu by pressing function key <F1>.

Select "Insert" from the softkey menu by pressing function key <F1>.

Enter the output to be looped back ("Q 0.5") and the delay for the positive and negative edges in the appropriate input fields.

Press <F7> "Info" for more information on each input field and on available editing options.

Exit the Input mode with <F6> "Store". The simulator stores the input data. In control system flowchart format, the looped-back IP 265 inputs/outputs are marked by "at" signs (@). (You can exit the Input mode without storing your entries by pressing <ESC>+<ESC>).

**Example:** Delete loopback to input 0.0

Move the cursor to input 0.0 in the control system flowchart's input bar.

Select "Loopback" in the function key menu by pressing <F1>.

Press <F2> in the new softkey menu.

If you pressed the right keys in the right menus, the loopback identifiers ("at" signs) should disappear from the control system flowchart.

Isolate parts of the control system flowchart (<F3> "Isolate")

Parts of the control system flowchart, all the way down to individual COM 265 language elements, can be isolated for simulation. Isolating points can serve a practical purpose when specific parts of the IP 265 user program (e.g. those containing errors) are not to affect other portions of the program.

In a control system flowchart, COM 265 inserts lower-case letters at the designated isolating points; these letters are then used to address the points.

Isolating points may be generated between language elements in all **binary** connecting lines. The connections (circuitry) are "cut" only in a "symbolic" or "graphic" sense.

Note

Isolating points may not be inserted in front of connectors or in connecting lines which lead to the control system flowchart's output bar.

The simulator treats the isolating point like an IP 265 input to which it can apply signal state "0" or "1" for the next program segment.

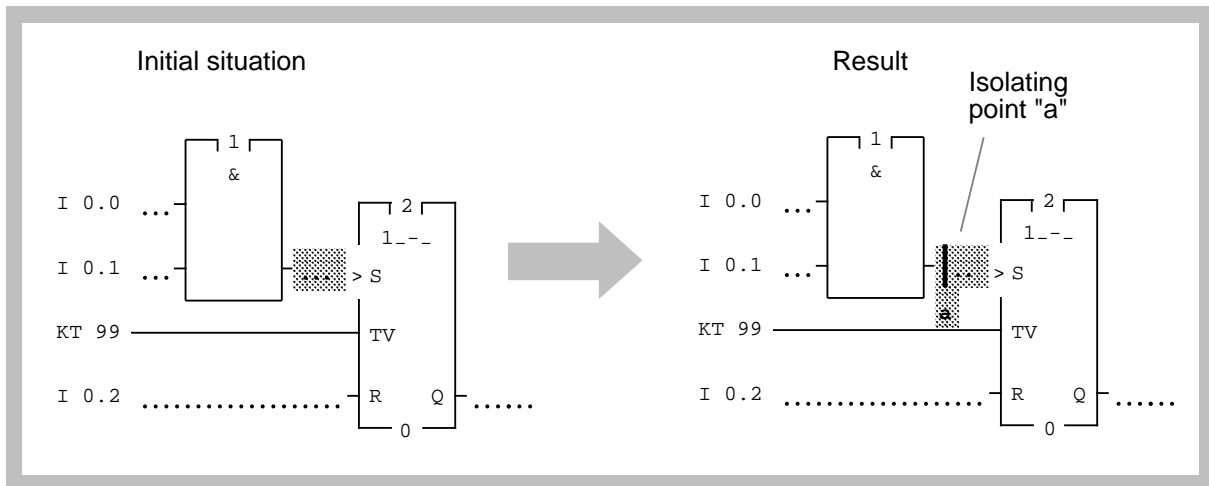
Note

Isolating points can be affected interactively by pressing the associated letter key.

Example: Generating an isolating point between an AND and a counter function

Move the cursor to the binary connecting line in which an isolating point is to be inserted.
 Select "Isolate" from the function key menu by pressing <F3>.
 Select "Insert" in the next menu level by pressing <F1>.

In control system flowchart format, the isolating point is marked with a separation symbol and a letter (for example "a").

**Example:** Deleting isolating point "a"

Move the cursor to the isolating point.
 Select the "Isolate" function in the function key menu by pressing <F3>.
 Select "Delete" in the next menu by pressing <F2>.

Specify actual values via function keys (<F2> "Value")

You can assign new actual values for retentive language elements and non-binary input parameters in the control system flowchart. You need not exit the Simulator level to do so.

The following manipulations are possible:

- New actual values for timers and counters
- New values for word input parameters for timers, counters and comparators
- Changing the status of an RS flipflop

All manipulations are displayed on the monitor screen.

Example: Changing the actual value of timer 1 from 50 ms to 100 ms

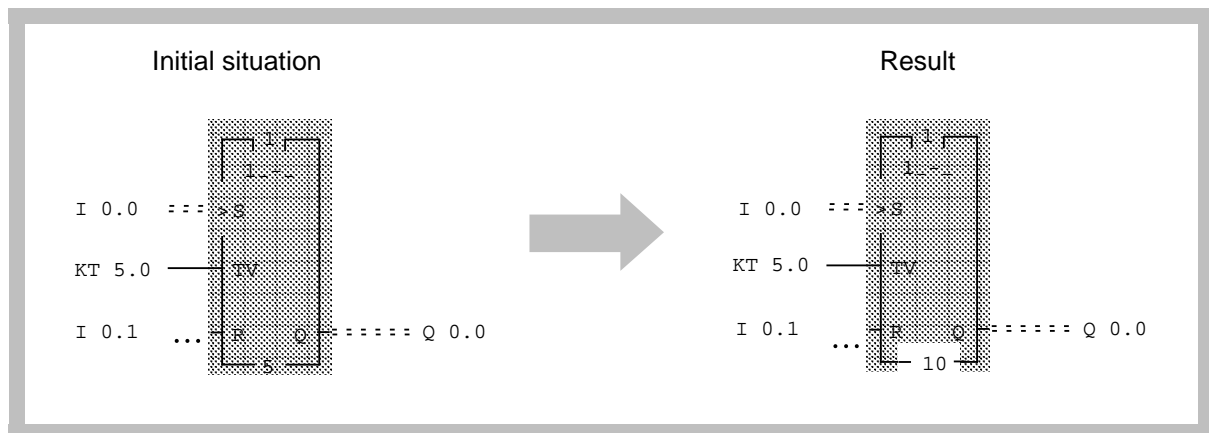
Set the cursor to the timer.

Select "Value" in the menu line by pressing function key <F2>.

The control system flowchart is faded out and replaced by the input field "New actual value".

Enter "10" for 100 ms.

Confirm your entry with <1> or <INSERT> .



The timer is set to "10" (100 ms) on the next IP 265 clock pulse.

Note

When manipulating timers, it is possible to change only the time (0 to 999). At the simulator level, it is not possible to change the time base.

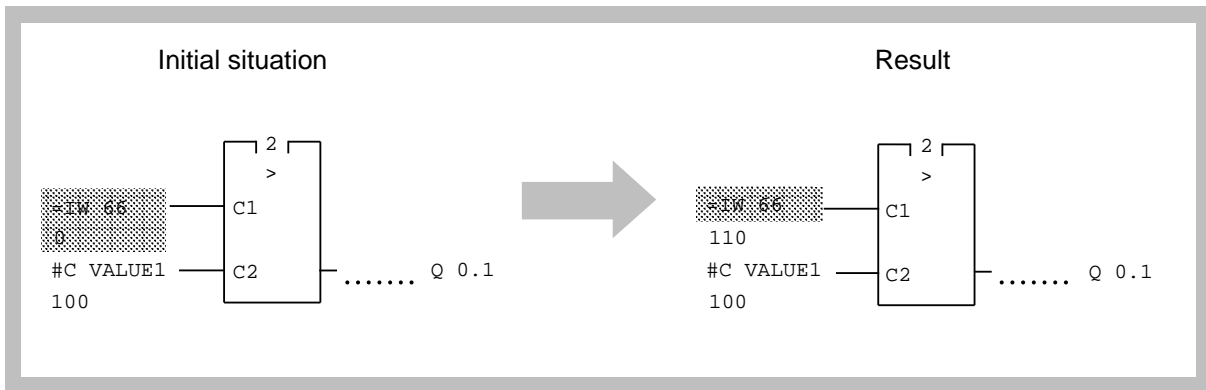
Remedy: Quick change to the Editor with <CORR>.

Example: A comparator compares a count (symbolic name: C VALUE1) with input word IW 66. The current actual value of input parameter IW 66 can be changed as follows:

- Move the cursor to the comparison value.
- Select "Value" in the function key menu by pressing <F2>.

The control system flowchart is replaced on the screen with the input field "New actual value".

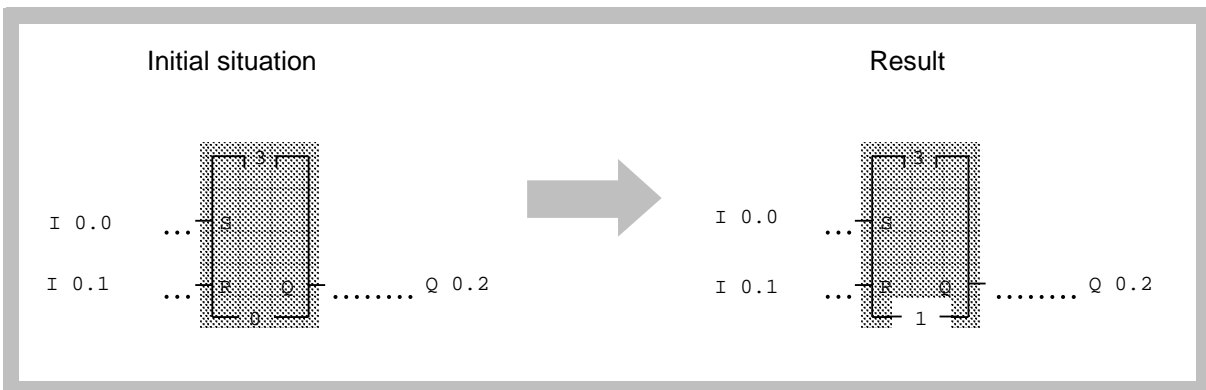
- Enter the new actual value (0 to 999) in the input field (e.g. 110).
- Confirm with <1> or <INSERT>.



The values "110" and "100" satisfy the compare condition ">". On the next IP 265 clock pulse, output Q 0.1 goes to "1".

Example: Changing the status of an RS flipflop

- Move the cursor to the language element.
- Select "Value" in the function key menu by pressing <F2>.



The next IP 265 clock pulse sets the RS flipflop. Output Q goes to "1".

Specify signal states

A signal state change from "0" to "1" may be specified for:

- IP 265 inputs and binary input parameters in the input bar
- Isolating points in the logic field

Table 10-2. Triggering a Signal State Change

Object	Signal state change can be triggered by:
IP 265 input (without loopback)	Possibility 1 (for direct and symbolic identifiers): Position cursor to identifier Press <F2> "Value"
	Possibility 2 (for direct and symbolic identifiers): Enter the direct address using the numeric keys. The IP 265 input need not be selected via the cursor. Example: The signal state of input I 0.5 is changed by entering <0>+<5>. Value range: <0>+<0> ... <2>+<3>
Looped-back IP 265 input	Position cursor to direct or symbolic identifier Press <F4> "Trigger"
Bit input parameter	Position cursor to direct or symbolic identifier Press <F2> "Value"
Isolating point	Press the key for the letter with which the isolating point is marked. The cursor need not be set to the isolating point. Value range: <a> to <z>

All of the above settings can be revoked by pressing the appropriate function key (Section 10.1.4).

Note

A change in the signal state of a looped-back IP 265 input via <F4> "Trigger" affects the control system flowchart only when the programmed delay time has expired.

Triggering timers and counters (<F4> "Trigger")

All retentive language elements can be triggered **within** the segments.

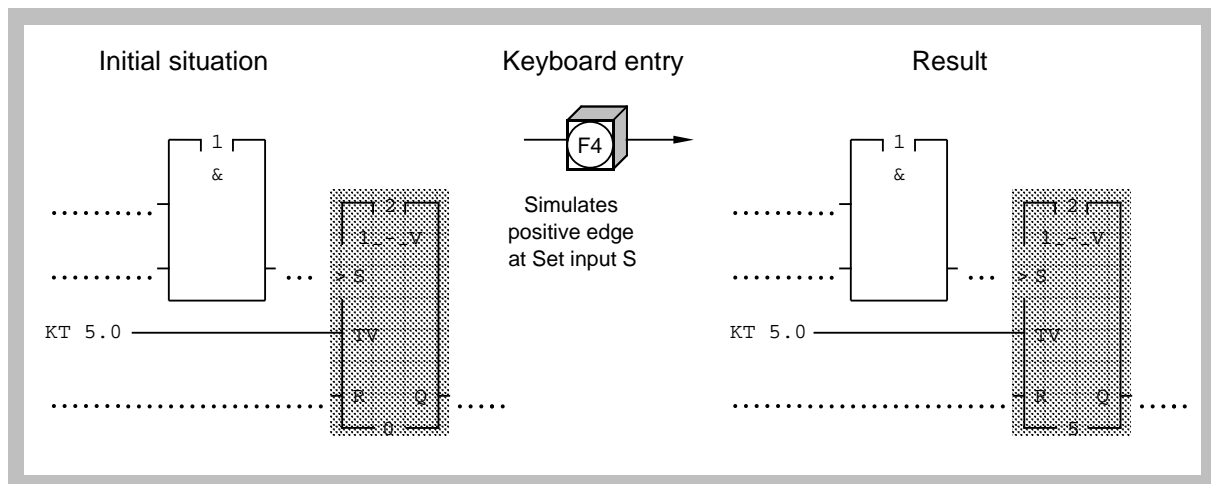
By pressing function key <F4> "Trigger" in the "Simulator" form's softkey menu, you can

- restart timers (the input conditions for S and R must be fulfilled if the timer is to be restarted on the next clock pulse, Table 9-8)
- restart counters
- affect loopbacks interactively ()

Example: Starting an extended pulse timer within a segment

Move the cursor to the language element.

Select "Trigger" in the softkey menu by pressing <F4>.



The timer is set to the current time "5" (50 ms), and started on the next IP 265 clock pulse.

10.1.4 Resetting Simulator Settings

Table 10-3. Resetting Simulator Settings

Resetting of:	User actions
<ul style="list-style-type: none">all simulator settings, including isolating points and loopbacksautomatic clocking to manual clocking	Select "Defaults" in the softkey menu by pressing <F6>. Press <F3> "Reset" in the next menu.
<ul style="list-style-type: none">IP 265 inputsinput parametersretentive language elements <p>Isolating points and loopbacks are retained. Automatic clocking of the simulator is restarted (if set).</p>	Select the "Reset" function in the softkey menu by pressing <F5>.

10.1.5 Symbols for Simulator Settings

All simulator settings are displayed on the monitor. The table below provides an overview of the simulator's symbols.

Table 10-4. Symbols for Simulator Settings

Symbol	Description	Example
• Signal state display at binary inputs and outputs		
...	Display: Signal state "0"	
—	Display: Digital flow	
===	Display: Signal state "1"	
• Signal state display at left and right of isolating points		
	Display: Left "1"/Right "0"	
	Display: Left "0"/Right "1"	
	Display: Left "1"/Right "1"	
	Display: Left "0"/Right "0"	
• Loopback to IP 265 inputs		
@@@@@@@@	IP 265 input to which an IP 265 output is looped back/ IP 265 output to which an IP 265 input is looped back	

Additional aids:

- All language elements in a control system flowchart are numbered. The numbers are faded onto the screen by the simulator.
- When digit keys (<00> to <23>) are used to change the signal state of IP 265 inputs, the first digit (byte address) is displayed at the upper right of the screen.
- Because the simulator is clock-controlled, the signal state changes at IP 265 inputs and isolating points and manipulations in actual values do not have an immediate effect. There is, in effect, an intermediate stage in which the new signal levels are present but have not yet been applied. The result of these manipulations do not become apparent until the next IP 265 "cycle".

The simulator displays signals which will take effect in the next IP 265 "cycle", using

- an exclamation point "!" to indicate a signal state change and
- hatching to indicate manipulation of actual values.

Table 10-5. Displaying Signal and Value Manipulations

Symbol	Description	Example
• Displaying signal and value manipulations		
!	Indicates a signal state change which will take effect in the next IP 265 "cycle"	
	Indicates the actual value of a timer, counter or flipflop which will take effect in the next IP 265 "cycle"	
	Indicates the actual value of an input parameter or output parameter which will take effect in the next IP 265 "cycle"	

10.1.6 Clocked Simulator Control

You can choose between

- manual clocking (default)
- automatic clocking

Manual clocking

You simulate the IP 265 clock pulse by pressing the space bar. Each depression of the space bar simulates an IP 265 cycle (7.8 μ s) (single-step mode).

Automatic clocking

You set the simulator cycle via function keys:

Select "Defaults" in the softkey menu by pressing <F6>.

Select "Sim Clock" in the next menu level with <F4>.

A selection window containing the following options is displayed on the monitor:

"Manual clocking" (default)
"Clock pulses/second"
"Seconds/clock pulse"

Move the cursor to the required text line and confirm with <1> or <INSERT>.

Enter the clocking info. Press <F7> "Info" for information on input fields and editing options.

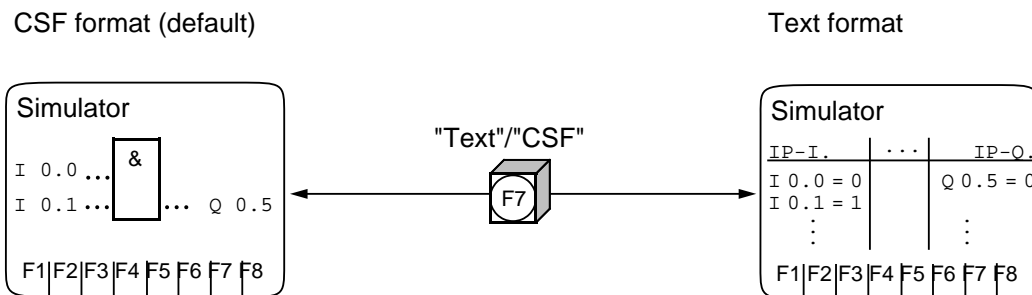
Press <1> or <INSERT> to confirm your entry.

The control system flowchart is returned to the screen. The real time, in μ s, and the current IP 265 cycle time are displayed at the upper right in the COM 265 screen form (Figure 10-1). When clocking is automatic, real time and cycle time are updated every second.

10.1.7 Text Representation

Text format is an alternative to control system flowchart format. The same simulator functions as in control system flowchart format are possible in text format; only the representation of the control system flowchart elements differs.

When the simulator is started, it is set for CSF format. A switch to text format can be initiated as follows:



In text format, all IP 265 inputs, IP 265 outputs and all parameters, with signal state or value, are automatically displayed in table form. You can also select relevant language elements in the control system flowchart. In text format, the simulator then displays these language elements, together with the function values.

Advantages of text format

Actual values and signal states of operands and language elements which are scattered over several segments in a control system flowchart are displayed in clear, concise form on a single screen page.

On the following pages, you will learn how to incorporate objects from a control system flowchart (COM 265 language elements) in text format.

Example: Incorporate language element "&" with position number "1" in text format

- Select CSF format (if necessary) with <F7> "CSF".
- Move the cursor to the language element "&" with position number "1".
- Select the "Defaults" function by pressing function key <F6>.
- Select "Object" in the next menu level by pressing <F1>.
- Change back to text format with <F7> "Text".

"Simulator" form

IP265 program: IP265		Segment:		Time: 0.000 ms		Clock: 0	
— IP inputs —		Input parameters		— IP outputs —		Output parameters -	
I 0.3	= 0	C VALUE1	= 100	Q 0.5	= 1		
I 0.6	= 1	C VALUE2	= 10	Q 0.7	= 0		
— Objects —		Objects		Objects		Objects	
&[1] = 0							
F1	F2	F3	F4	F5	F6	F7	F8
Loopback	Value		Trigger	Reset	Defaults	CSF	Return

Figure 10-2. "Simulator" Form; Text Format

10.2 On-Line Testing of the IP 265

The logically correct IP 265 user program can be tested on-line, i.e. in the IP 265 (under CPU control and with process I/Os). COM 265 provides a special test function to support on-line testing.

The "On-line test" function makes it possible for testing of the IP 265 program to be controlled and monitored via the programmer. You can

- forward control commands (RUN, STOP) to the IP 265 and read back status info from the IP 265
- influence input parameters and monitor output parameters.

Note

You can also monitor the local IP 265 inputs and outputs on the programmer screen when they are interconnected as output parameters in the IP 265 user program.

Note

During an on-line test, the CPU (CPU user program) and COM 265 use the same address space (PIQ/PII) to access the IP 265.

Prerequisites for an on-line test

- The IP 265's slot must have been set in COM 265's "Defaults" form
- The IP 265 user program (control of high-speed subprocess) must have been loaded in the IP 265 via the I/O bus.
- The CPU user program (which controls the entire process) must have been loaded into the CPU, and the CPU must be in RUN mode.

Starting an on-line test

Press the function key for "Online".

A table containing current parameter values is displayed on the monitor.

"Online" form

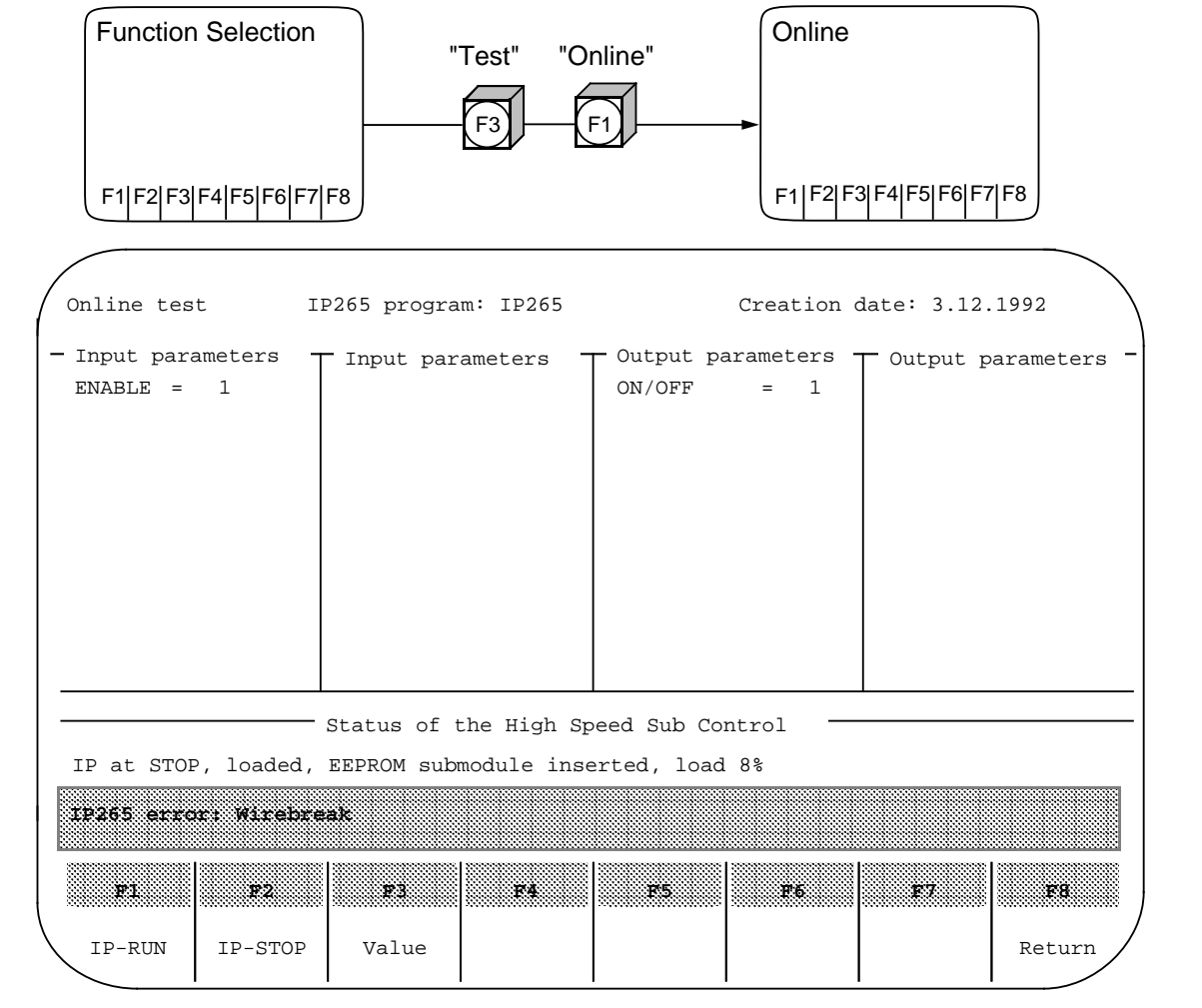


Figure 8-3. "Online" Form

COM 265 supports on-line testing via function keys. Press <HELP> for information on the execution of on-line test functions.

User displays

- Parameter values
The parameter values are displayed in table form, by word or by bit. Parameters with symbolic identifiers in the IP 265 user program also appear in the on-line test parameter list under these symbolic identifiers.

Note

If the parameters with symbolic addresses are to be displayed on the programmer screen, the same program must be in the programmer's current drive.

- Status information
A text line containing the following status messages is displayed beneath the parameter list:
 - Operational status of the IP 265 (STOP, RUN)
 - Load status of the IP 265 (loaded, not loaded)
 - Type of memory submodule plugged into the IP 265 (EPROM or EEPROM)
- IP 265 capacity utilization
The load on the IP 265's program capacity, in percent.

The parameter values and the status info are updated cyclically. Updating is much slower than IP 265-internal processing (IP 265 clock pulse), i.e. high-speed signal changes cannot be followed on the monitor screen.

Possible user actions

You can start or stop program scanning easily via function keys <F1> and <F2>.

You can also manipulate the actual values for bit and word input parameters:

Changing bit parameters:

Move the cursor to the bit input parameter.

Press <F3>.

This changes the signal state from "0" to "1" or vice versa.

Changing word input parameters:

Move the cursor to the word input parameter.

Press <F3>.

An input field for the actual value is displayed on the monitor screen

Enter the new actual value and store your entry in the parameter list with <1> or <INSERT>.

Note

Since the CPU and COM 265 (programmer) both access the IP 265 via the same address space, the CPU user program may, under certain circumstances, overwrite COM 265 data in the PIQ.

Error display

COM 265 displays textual error messages in on-line mode (e.g. "Wirebreak"). Error texts are displayed in a Help window.

10.3 Wiring Test

COM 265 provides a function for testing the electrical continuity between the IP 265's inputs/outputs and the process I/Os' sensors or actuators of the process I/O.

There is no "wiring test" function as such. Instead, there is a special IP 265 utility called "WIRE", which must be loaded into the IP when you want to perform a wiring test. The "On-line" function is used to display the results of the wiring test.

The WIREPIP.S5D file is on the COM 265 installation floppy.

Preparations for the wiring test

Select the "WIRE" test.

There are two ways of "Selecting" the "WIRE" utility:

1. Press <F2> "Wiring" in the "Test" form and confirm with <1>. The program name "WIRE" is then automatically displayed in COM 265's "Defaults" form.
2. Enter the program name "WIRE" in the "Defaults" form.

Load the "WIRE" utility into the IP 265 and set the IP 265 to RUN mode via the "Load IP 265" service function.

Start the wiring test. Invoke the "On-line" test function (Section 10.2).

The following are displayed in table form and updated cyclically:

- IP 265 24 V inputs
- IP 265 24 V outputs
- IP 265 5 V differential inputs

Note

The "WIRE" program does not have to be compiled.

Note

If you load the "WIRE" program into the IP 265 via the I/O bus, remember that you will also overwrite the IP 265's EEPROM memory submodule, if one is plugged in.

Testing the electrical continuity of the outputs

Move the cursor to the relevant output.
Press <F3> "Value".
The signal state changes from "0" to "1".
Observe the responses of the actuators of the process I/O.

Testing the electrical continuity of the inputs

Initiate a signal state change by de-energizing one of the actuators of the process I/O.
Observe the responses of the signal state display for the relevant input on the programmer screen.

11 COM 265 Services and File Functions

11.1	Services	11- 1
11.1.1	Invoking the Symbols Editor	11- 2
11.1.2	Loading the IP 265 Via the I/O Bus	11- 2
11.1.3	Storing an IP 265 User Program on a Memory Submodule	11- 4
11.1.4	Reading an IP 265 User Program from a Memory Submodule	11- 6
11.1.5	Printing an IP 265 User Program	11- 8
11.2	File Functions	11- 9
11.2.1	Copying an IP 265 User Program	11- 10
11.2.2	Deleting an IP 265 User Program	11- 11
11.2.3	Display Directory of All IP 265 User Programs	11- 12
11.2.4	Renaming an IP 265 User Program	11- 13

Figures

11-1	"Services" Menu	11- 1
11-2	"Load IP 265 via I/O Bus" Form	11- 3
11-3	"E(E)PROMs" Form; Blowing E(E)PROM	11- 5
11-4	"E(E)PROMs" Form; Reading E(E)PROMs	11- 7
11-5	"Print" Form; Printing the IP 265 Program	11- 8
11-6	"File Functions" Menu	11- 9
11-7	"Copy" form	11- 10
11-8	"Delete" Form	11- 11
11-9	"Directory" Form	11- 12
11-10	"Rename" Form	11- 13

11 COM 265 Services and File Functions

COM 265 provides a number of services and file functions to support IP 265 startup.

11.1 Services

COM 265 services are used to

- invoke the symbols editor (local IP 265 assignment list)
- initiate output of the IP 265 user program to various media

"Services" menu

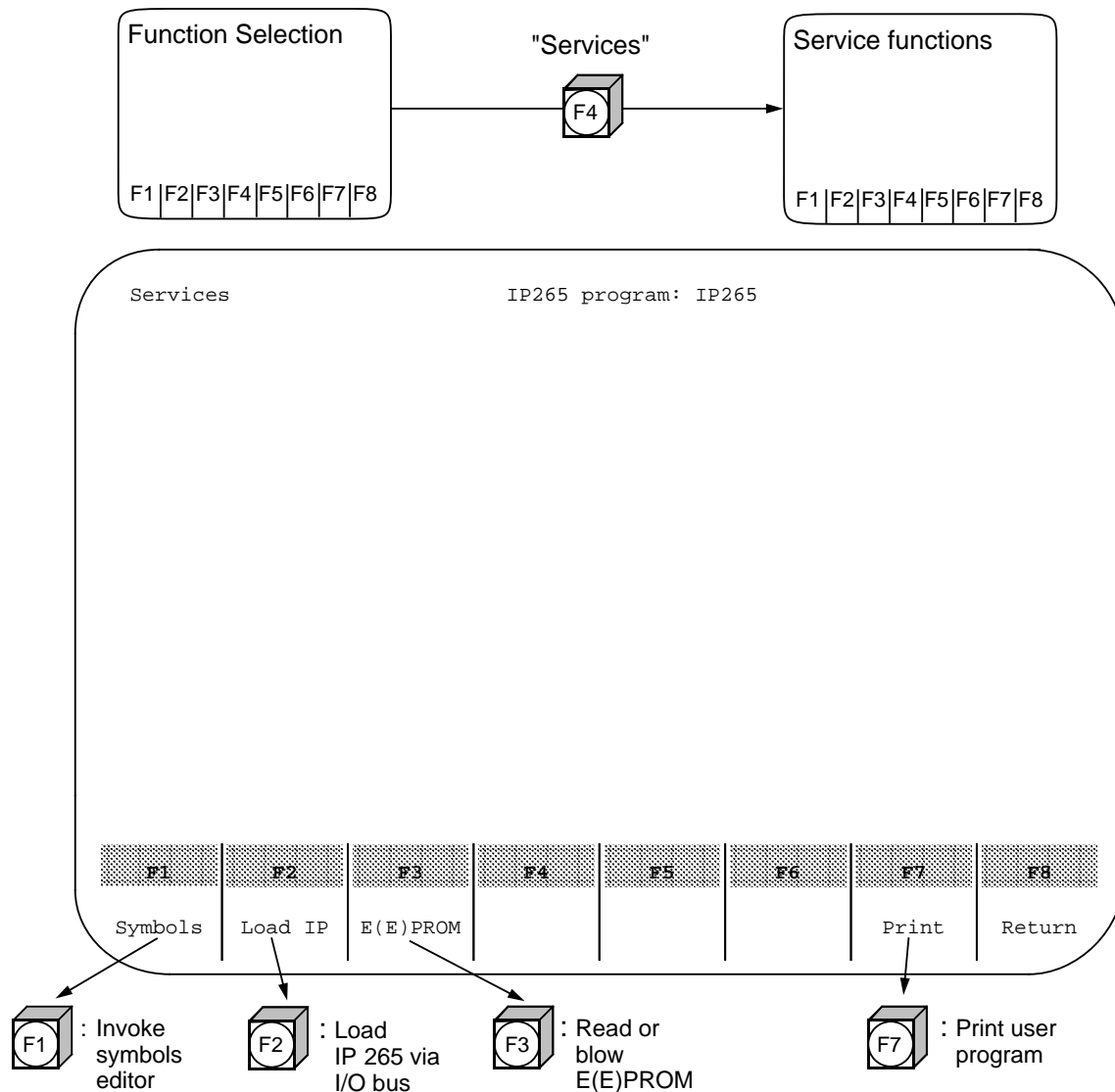


Figure 11-1. "Services" Menu

11.1.1 Invoking the Symbols Editor

COM 265 is equipped with a special symbols editor which makes it possible to enter operand commentary and symbolic operand identifiers from the IP 265 user program in an assignment list.

You will find detailed information on reorganizing the operand references in the assignment list and removing operands from the list in Section 9.4.2, "Entries in the Local IP 265 Assignment List".

11.1.2 Loading the IP 265 Via the I/O Bus

For on-line testing, the IP 265 user program generated on the programmer can be loaded directly into the IP 265. The load data for the user program is transferred over the CPU's PIQ and the external I/O bus to the IP 265. To enable this, the programmer must be interfaced to the PLC (CPU).

Note

Only an IP 265 user program can be loaded into the IP 265. When a new user program is loaded into the IP, it overwrites the "old" user program.

Prerequisites for starting the load procedure

- The programmer's program memory must contain an IP 265 user program that was generated with COM 265 and compiled without errors.
- The IP 265 submodule receptacle must either be empty or contain an EEPROM memory submodule.
- The following must have been entered in the COM 265 "Defaults" form:
 - The current slot of the IP 265 to be loaded (e.g. "0")
 - The filename of the IP 265 user program to be loaded into the IP 265 (e.g. "IP265")
- Because the I/O bus is serviced by the CPU, additional user actions may be necessary, depending on the operational status of the CPU **prior to** COM 265 servicing. COM 265 provides adequate support through appropriate messages and directives and prompts.

Note

COM 265 and the CPU both have access to the PIQ in CPU RAM. During loading, you must ensure that no control commands or input parameters can be written by the CPU user program to the PIQ (Section 6.1, "Control of IP 265 Program Scanning").

Starting the load

"Load" form

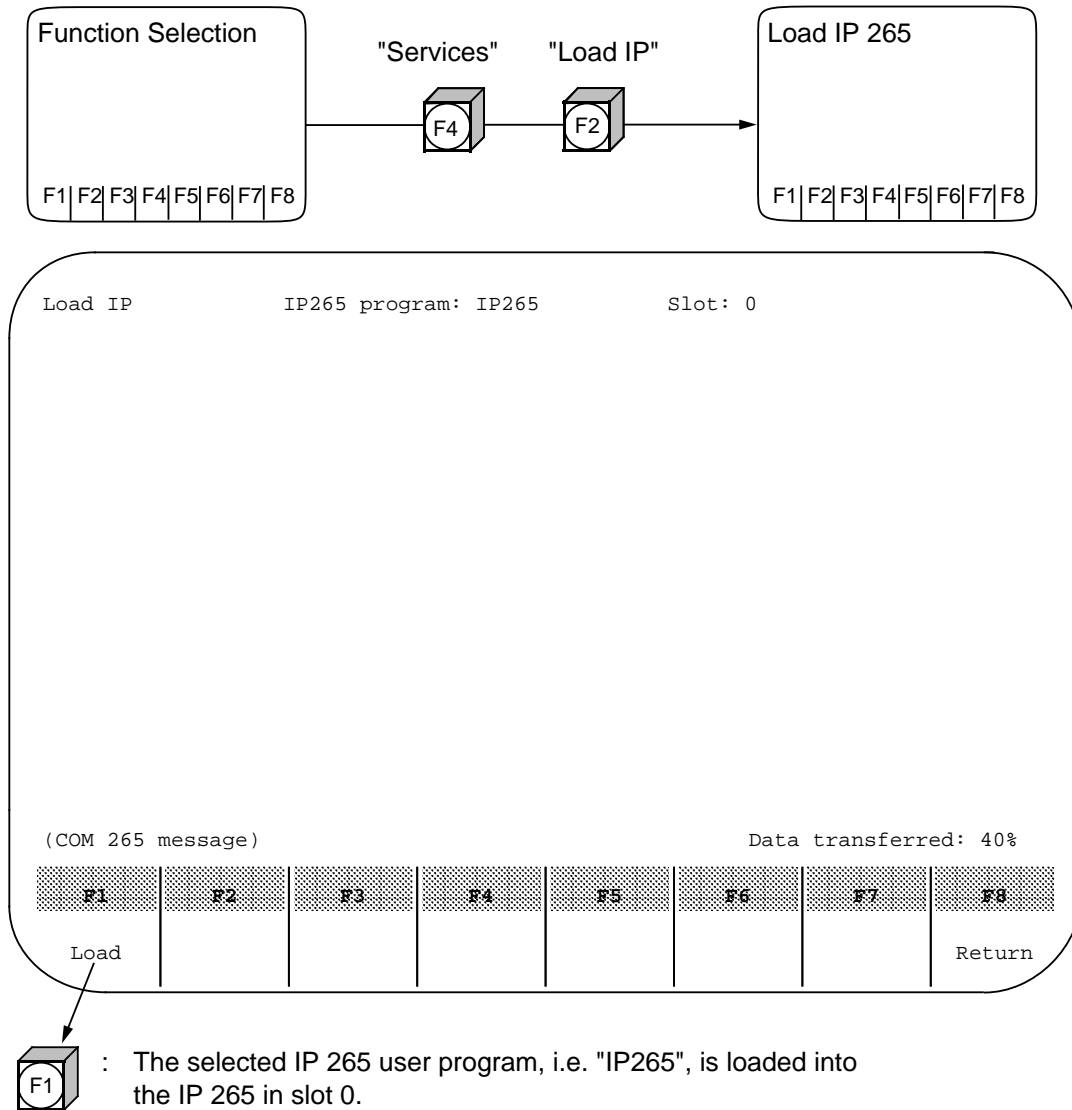


Figure 11-2. "Load IP 265 via I/O Bus" Form

Once the load has been started, COM 265 takes care of loading the program data. COM 265 displays the amount of program data loaded, in percent, on the message line at the right of the screen.

Note

If the IP 265 is equipped with an EEPROM, the user program is loaded into this submodule as well.

LED indicators

The STOP LED on the module's frontplate flashes at high frequency to indicate that the module is being loaded over the I/O bus.

Duration of the load procedure

The IP 265 user program is transferred in 1000 data cycles. At an average CPU cycle time of 20 ms, the load would take approximately 20 s. If an EEPROM has to be overwritten, the load takes approximately 80 s.

11.1.3 Storing an IP 265 User Program on a Memory Submodule

The IP 265 user program can be programmed direct into a memory submodule plugged into the programmer. COM 265 provides a special service for this purpose.

Note

The STEP 5 package "EPROM/EEPROM" **cannot** be used to store an IP 265 user program on an E(E)PROM memory submodule!

Prerequisites for a successful storage procedure:

- The IP 265 user program to be stored (e.g. "IP265") was selected in COM 265's "Defaults" form.
- The IP 265 user program (e.g. "IP265") was compiled without error by the COM 265 compiler.
- The programmer is equipped with an EEPROM memory submodule or an empty EPROM memory submodule.

You will find a list of permissible E(E)PROM memory submodules in Appendix G, "Accessories and Order Numbers", of the Product Manual.

Starting the store

"E(E)PROMs" form

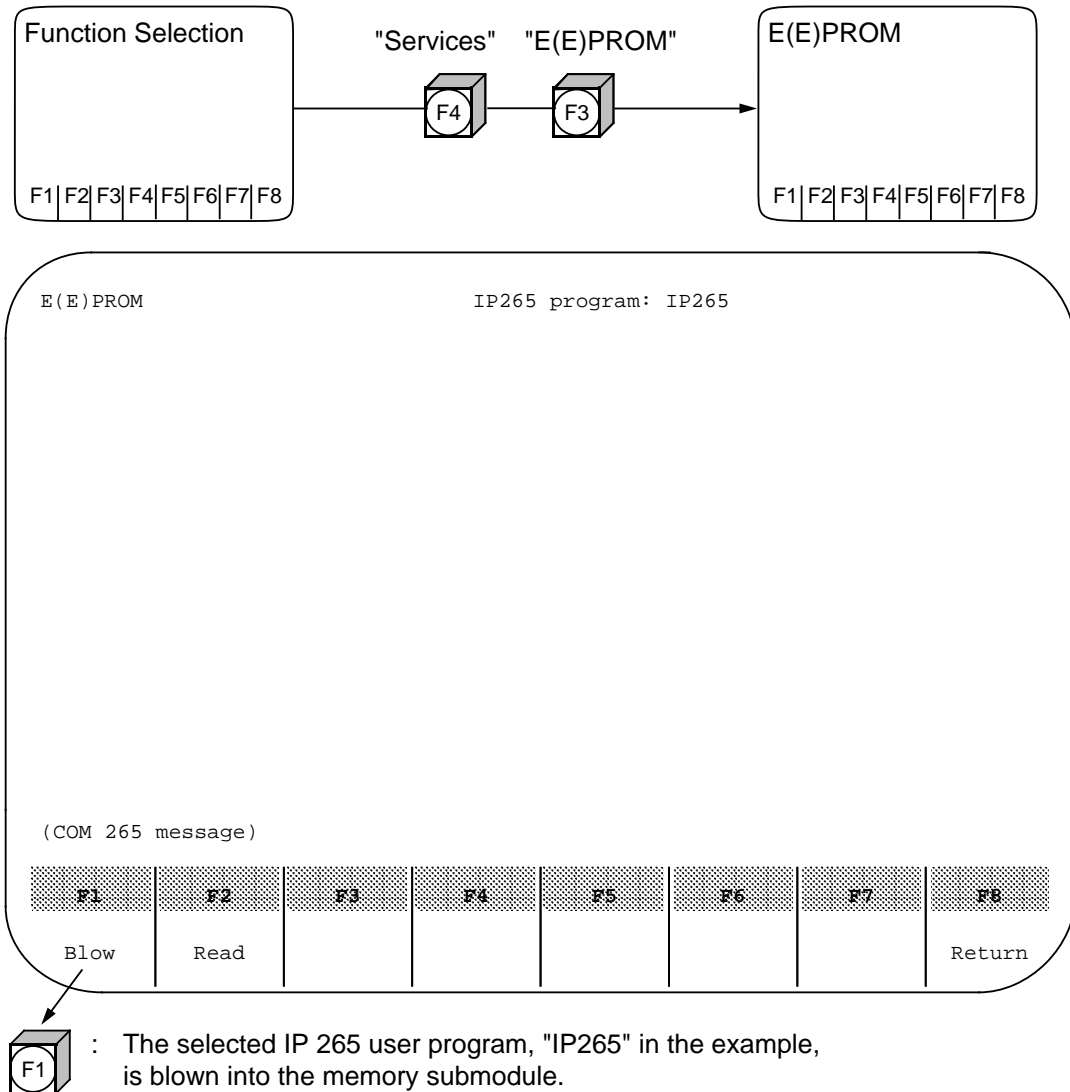


Figure 11-3. "E(E)PROMs" Form; Blowing E(E)PROMs

Once the service function has been started, COM 265 takes over storing of the program data.

Note

Operands and segment comments are not stored on the memory submodule.

11.1.4 Reading an IP 265 User Program from a Memory Submodule

Once you have stored an IP 265 user program in an E(E)PROM, you can display it on the programmer screen any time you wish.

For this purpose, COM 265 provides a special service function for reading the data stored on E(E)PROM into the programmer's program memory. Subsequent selection of the "Editor" form will screen the control system flowchart for that IP 265 user program.

Note

The STEP 5 package "EPROM/EEPROM" **cannot** be used to read an IP 265 user program from an E(E)PROM memory submodule.

Note

If the IP 265 user program is overwritten on the programmer during loading, operands and segment comments are lost.

Prerequisites for reading the memory submodule

- An E(E)PROM submodule containing the IP 265 user program to be read must be plugged into the programmer.

Starting the read procedure

"E(E)PROMs" form

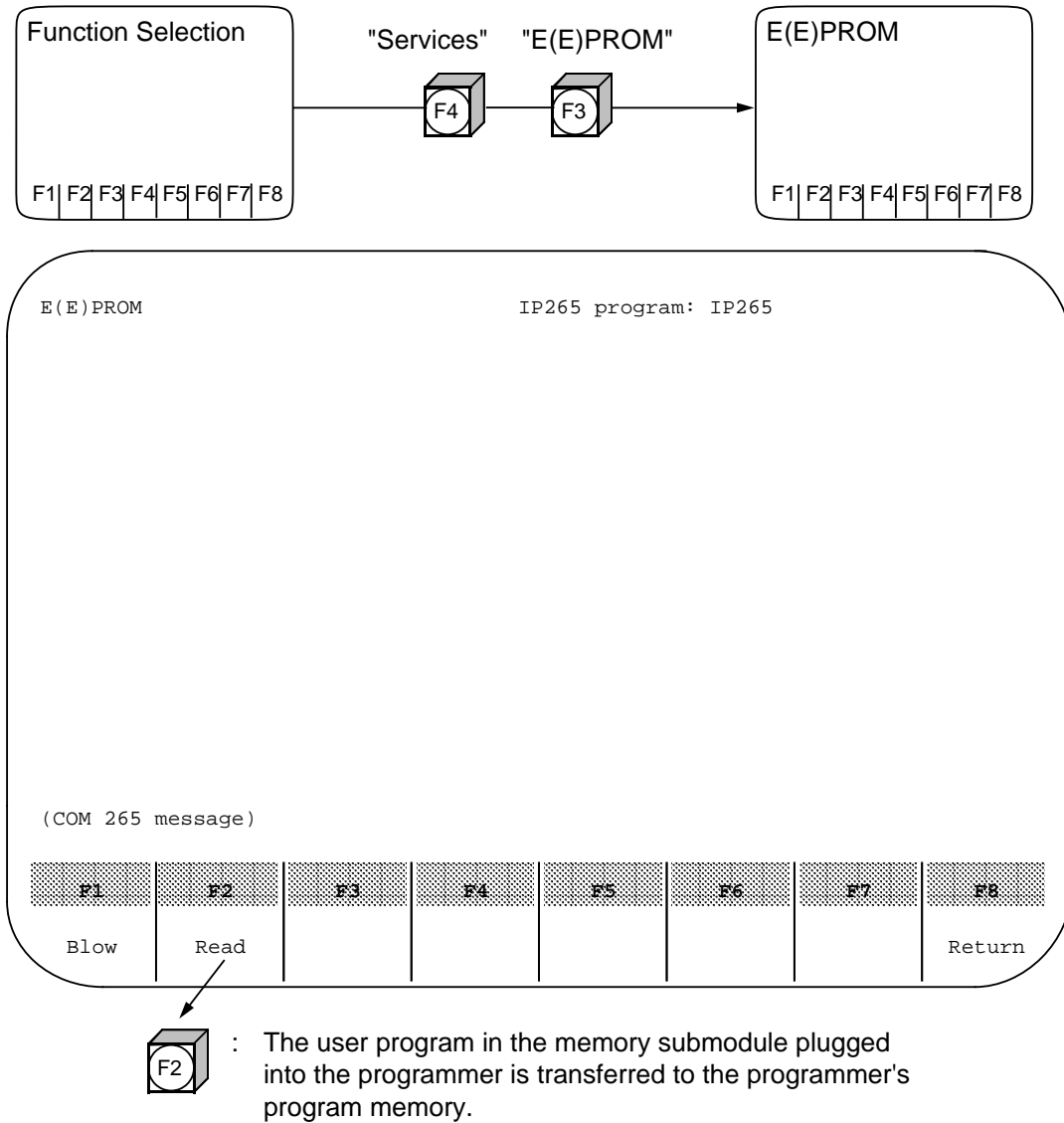


Figure 11-4. "E(E)PROMs" Form; Reading E(E)PROMs

Once the service function has been started, COM 265 handles the operation autonomously.

Note

The IP 265 user program read from the E(E)PROM is the current program for which all COM 265 functions can now be used.

11.1.5 Printing an IP 265 User Program

The "Print" service outputs the following to the printer interfaced to the programmer:

- All segments in the IP 265 user program (CSF format), with segment names
- All lines in the operand assignment list for those segments
- Segment commentary for all segments, if any
- PLC footer file, if available and if selected

The PLC printer file specified in the "Defaults" form, if any, will also be taken into account.

Note

The PLC footer file and the PLC printer file are external files, and can be programmed with the aid of the STEP 5 package.

Prerequisites for printing

- The following must be specified in COM 265's "Defaults" form:
 - The file name of the IP 265 user program to be printed out
 - The file name of the PLC footer file, if any
 - The file name of the PLC printer file, if any

Starting the printout

"Print" form

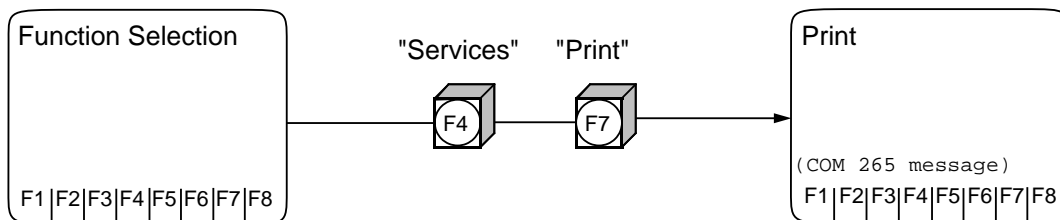


Figure 11-5. "Print" Form; Printing the IP 265 Program

11.2 File Functions

The IP 265 user program is stored at the MS-DOS operating system level in a file named "xxxxxPIP.S5D". Only the "xxxxx" part of the filename (max. 5 characters) appears in the COM 265 screen forms. This portion of the name, referred to in the manual as the program name, is chosen when the IP 265 user program is written. When executing a COM 265 file function, the user programs are referred to by their program names.

The COM 265 file functions can be used to

- rename, copy or delete an IP 265 user program
- display the directory of IP 265 user programs

"File functions" menu

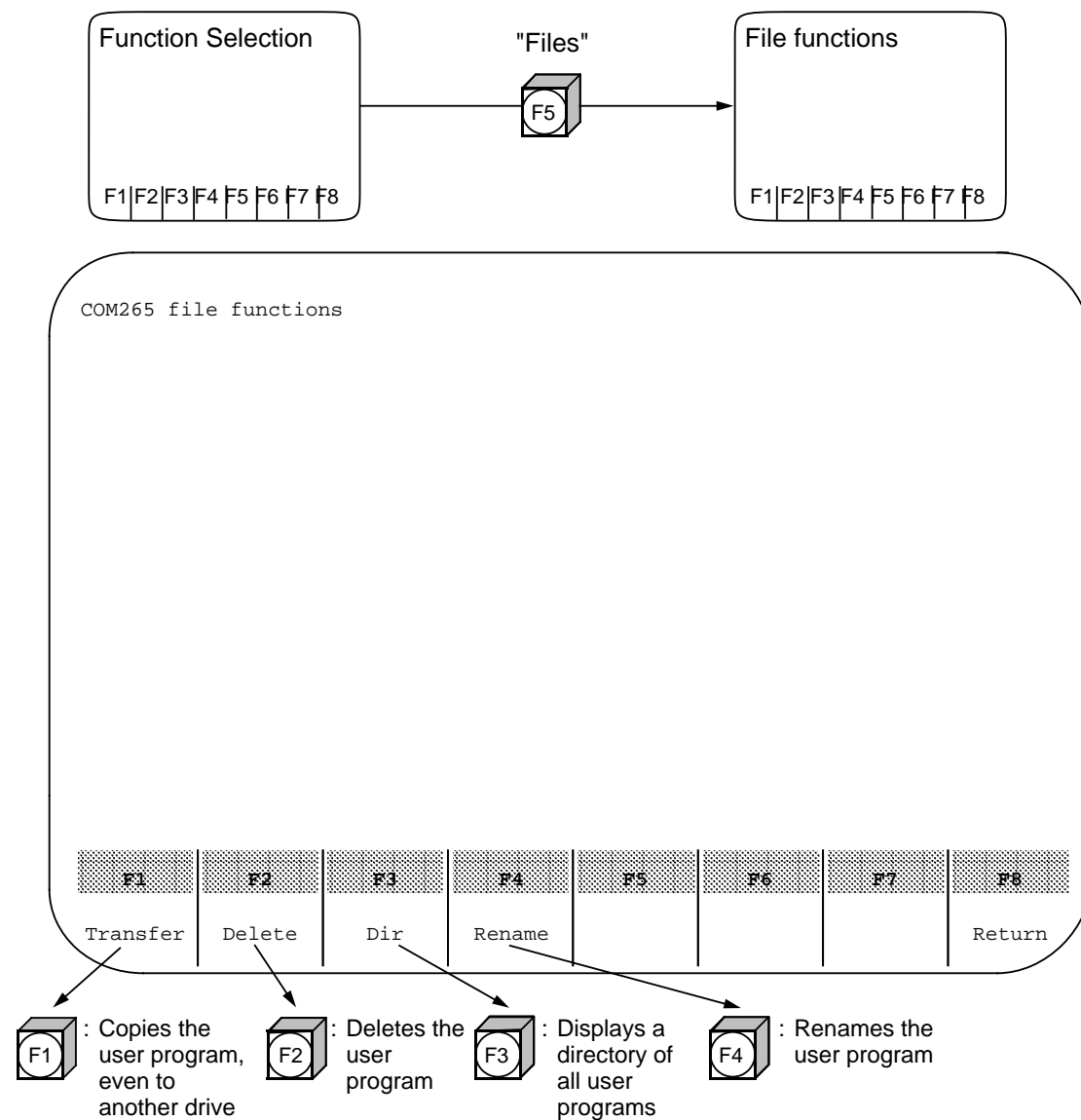


Figure 11-6. "File Functions" Menu

11.2.1 Copying an IP 265 User Program

An existing IP 265 user program can be copied to another drive for program management purposes.

Parameters for the copy function

"Copy" form

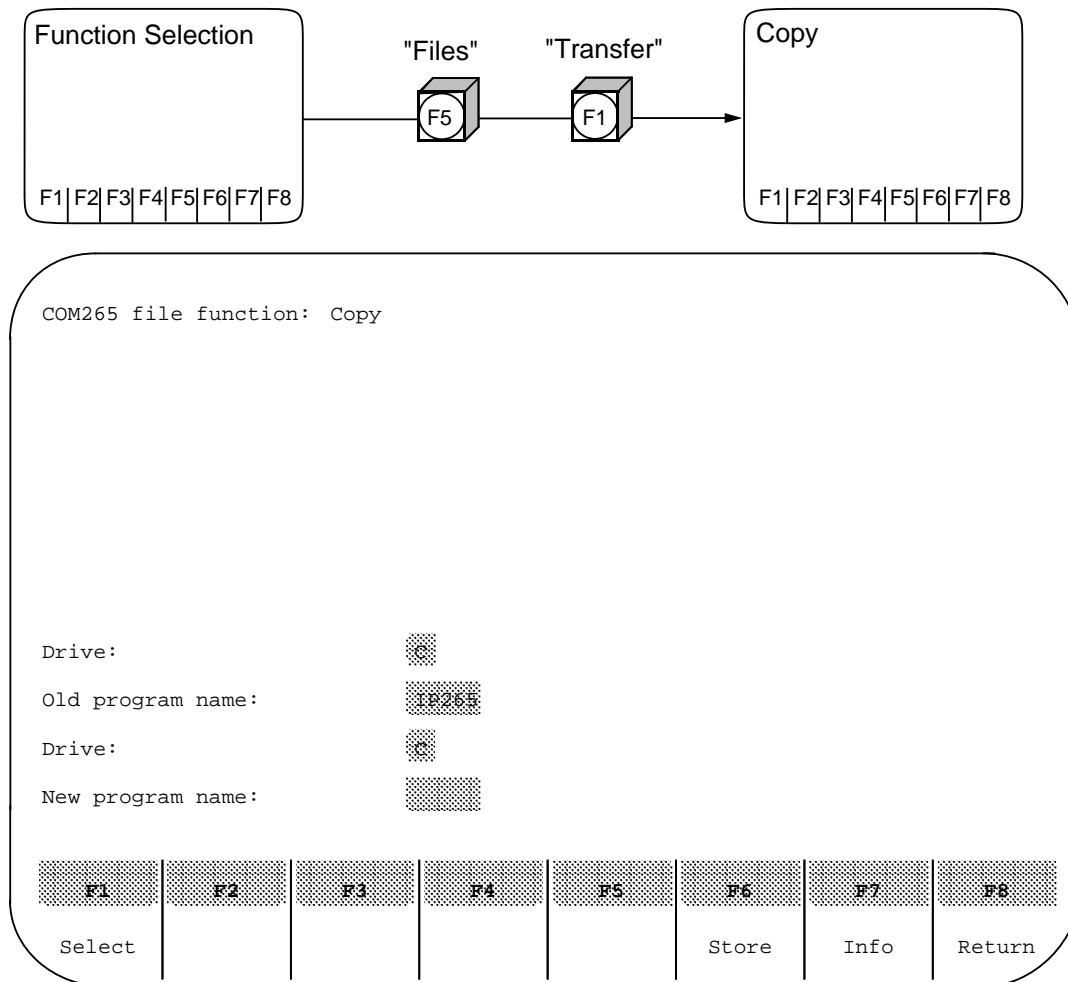


Figure 11-7. "Copy" Form

Move the cursor to each input field in succession, entering the appropriate program name or drive in each field. Press <F7> "Info" for information on input fields and editing options. Confirm your entries with <1> or <INSERT>.

Starting the copy function

Start the copy with <F6> "Store". (The copy can be aborted by pressing <F8> "Return")

11.2.2 Deleting an IP 265 User Program

You can delete IP 265 user programs individually.

Parameters for the delete function

"Delete" form

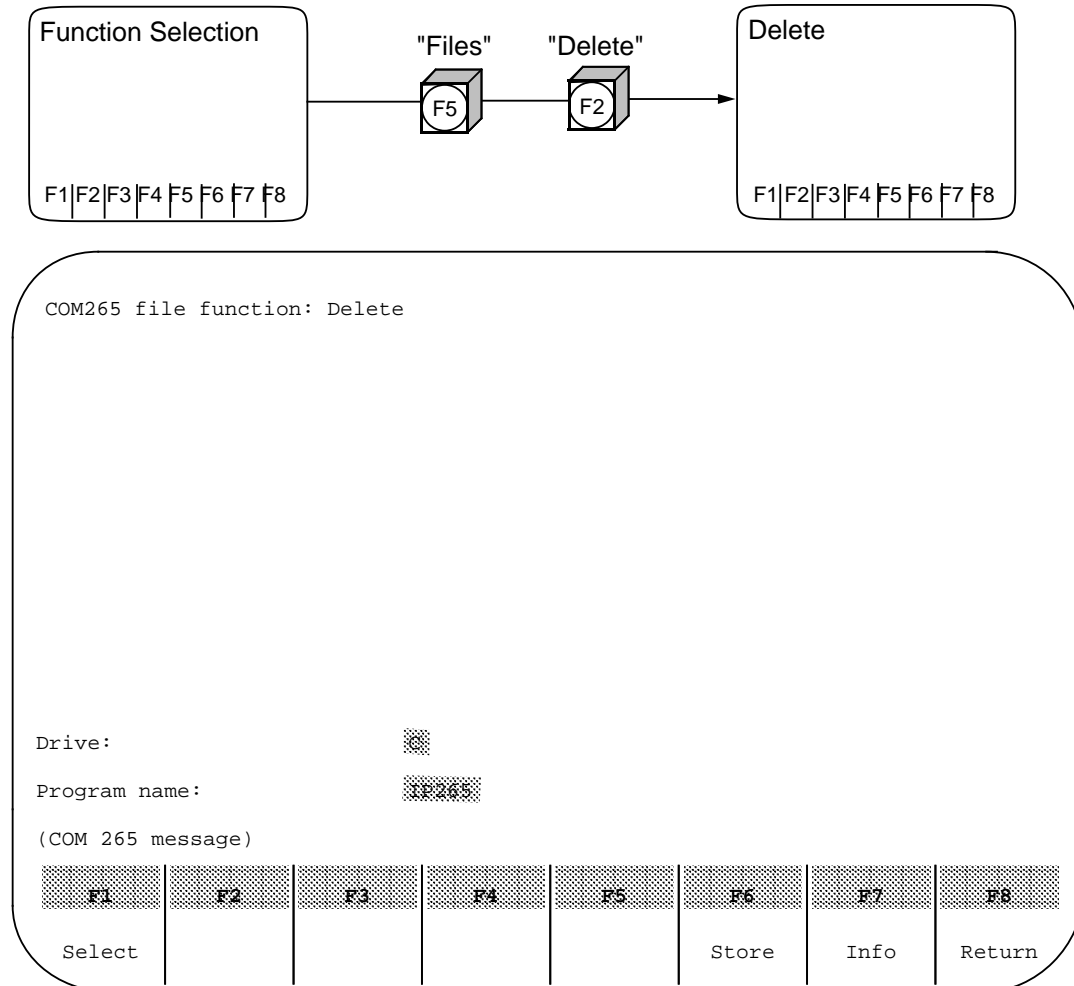


Figure 11-8. "Delete" Form

Move the cursor to each input field in succession, entering the drive and program name in the appropriate field. Press <F7> "Info" for information on input fields and editing options. Confirm your entries with <1> or <INSERT> .

Starting the delete function

Start the delete with <F6> "Store".

Press <1> or <INSERT> to execute the delete function; press <ESC> to exit the "Delete" function prior to its execution.

11.2.3 Display Directory of All IP 265 User Programs

If you wish, you can output a directory of all IP 265 user programs.

Starting the display function

"Directory" form

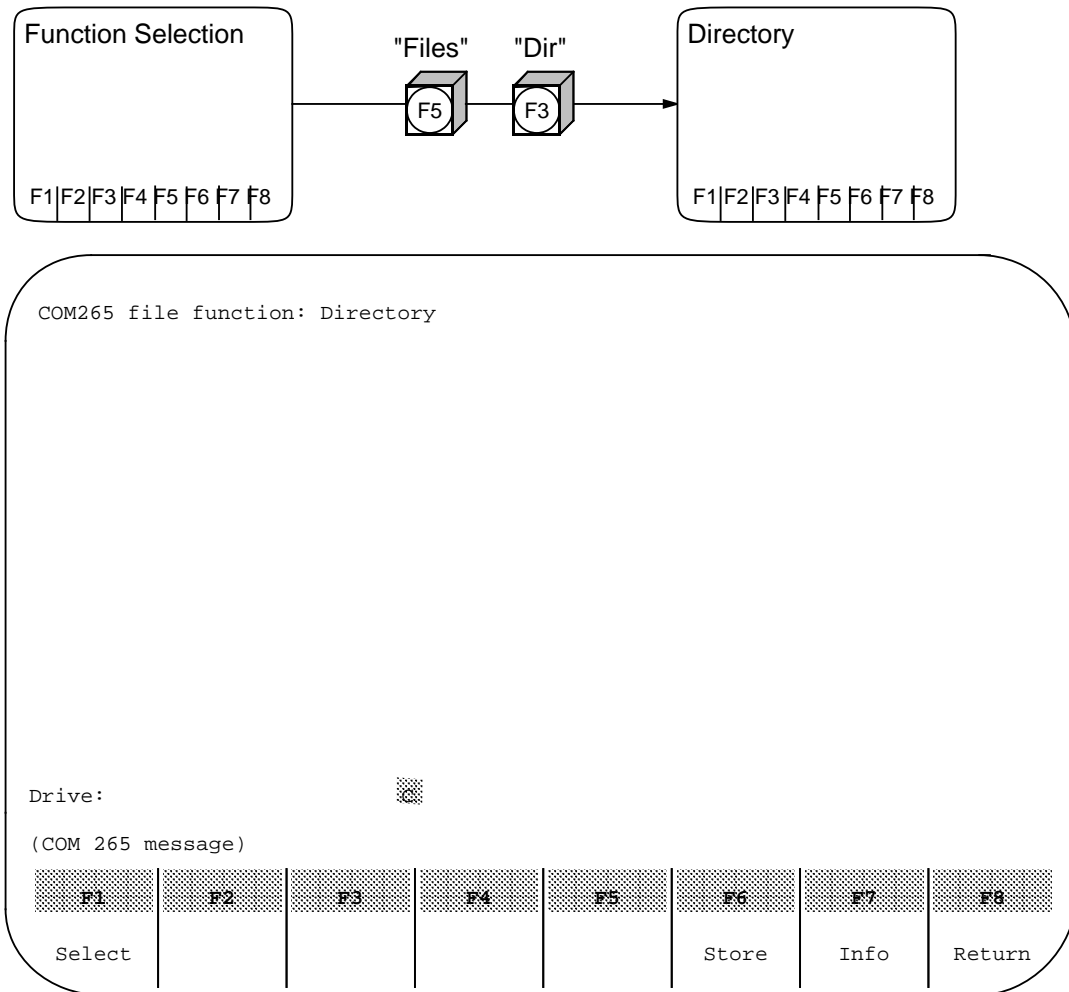


Figure 11-9. "Directory" Form

Move the cursor to the input field and enter the drive. Press <F7> "Info" to screen information on the input field and editing options.
Confirm your entries with <1> or <INSERT>.

Starting the directory display

Start the directory display by pressing <F6> "Store".

11.2.4 Renaming an IP 265 User Program

The name of an existing IP 265 user program can be changed for program management purposes.

Parameters for the rename function

"Rename" form

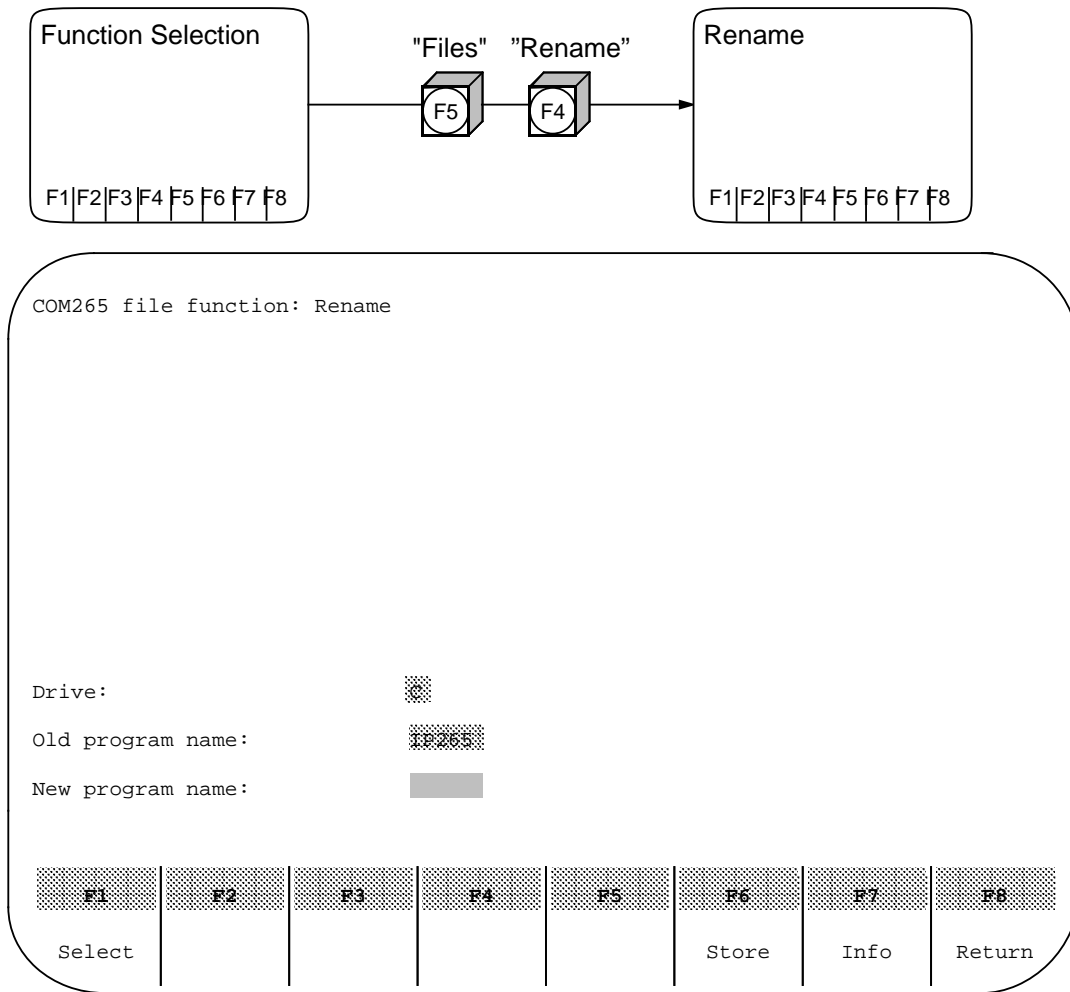


Figure 11-10. "Rename" Form

Move the cursor to each input field in succession and make the appropriate entry in each field. Press <F7> "Info" to screen information on the input fields and editing options. Confirm your entries with <1> or <INSERT>.

Starting the rename function

Start the function by pressing <F6> "Store".

12 IP 265 Expansion		
12.1	General Remarks	12 - 1
12.2	Interconnecting Two IP 265s	12 - 5
12.3	Power Supply for Expanded IP 265s	12 - 5
12.4	Addressing the Expansion Inputs/Outputs	12 - 6
12.5	Sample Program	12 - 6

Figures		
12-1	Data Transfer Between CPU and Two Expanded IP 265s	12 - 2
12-2	Simplified Diagram for Non-Isolated Connection of an Expanded IP 265 to the PLC	12 - 5
12-3	Process: Fast Shutdown in the Event of a Malfunction	12 - 7
12-4	Interfaces Used on the Two IP 265s in the Sample Application: "Fast Shutdown in the Event of a Malfunction"	12 - 8
12-5	Structure of the CPU Program Section for Controlling Two Expanded IP 265s	12 - 10
Tables		
12-1	Addresses of the Expansion Inputs/Outputs	12 - 6

12 IP 265 Expansion

The information presented in this section deals only with the particularities of interfacing two IP 265s. General knowledge regarding operation and startup of the IP 265 have been assumed (Sections 2 to 11).

12.1 General Remarks

What is meant by "expanding" an IP 265 ?

An IP 265 can be interfaced to a second IP 265 over a special interface (called an expansion interface). In this way, a second IP 265 can postprocess an IP 265's outputs and intermediate results. The processing width of the expansion interface is 8 bits.

When is expansion necessary?

Expansion may become necessary when

- the number of digital 24 V inputs and 24 V outputs on **one** IP 265 or
- the program capacity of **one** IP 265

is insufficient to control a high-speed subprocess.

What does expansion mean when it comes to writing user programs?

The IP 265 user program is divided between the two interconnected IP 265s. The programs (or program segments) are recombined into a single IP 265 user program over the special expansion interface.

Note

It is not possible to use both an expansion interface and IP 265 differential inputs at the same time.

Dynamic response

- Over the expansion interface, a signal is delayed by 7.8 μs .
- The CPU user program uses the external I/O bus for error control and recovery and for STOP/RUN control of both IP 265s. Since control word and status word for both IP 265s are forwarded cyclically in one CPU data cycle, there is no delay when e.g. both IPs are to be set to STOP.

Note

The expansion interface has no STOP or error signal line of its own.

- Parameter initialization for both IP 265s is also synchronous. Because input parameters and output parameters are forwarded cyclically in one CPU data cycle, there is no delay in initializing the two IPs.

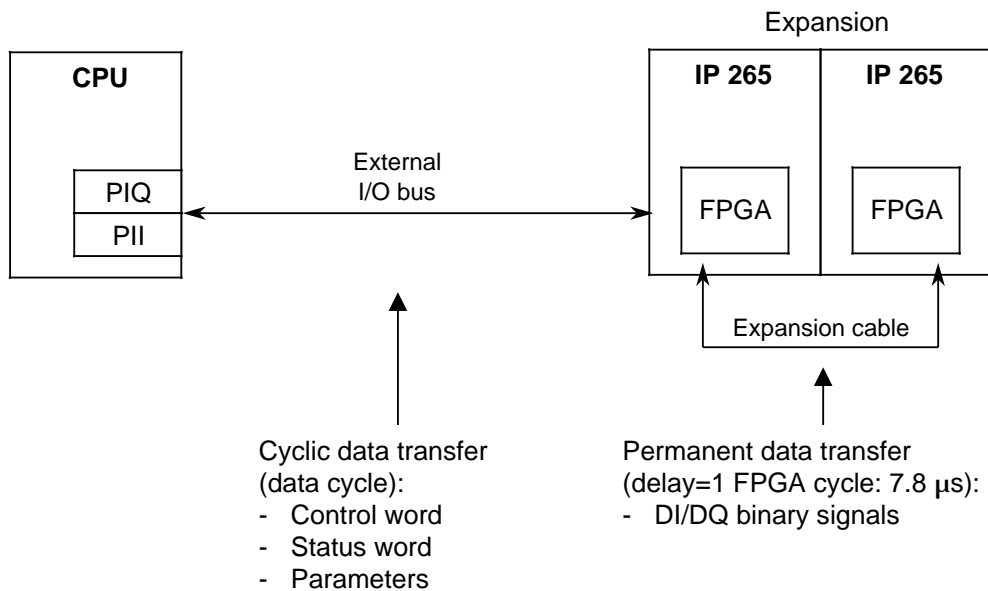


Figure 12-1. Data Transfer Between CPU and Two Expanded IP 265s

Response to a change in the operational status

STOP and RUN commands from the CPU reach expanded IP 265s without any delay.

In order to ensure error-free, reproducible control of both IPs, you should follow the guidelines below in the event of an IP 265 operational status change:

- The CPU should set both IPs to RUN or STOP at the same time.
- The programmed response to a CPU STOP should be identical for both IPs.
- The CPU must query the operational states of both IPs continually, and match them if necessary. If one IP has gone to STOP because of a fault or error, the CPU must set the other to STOP as well.

Note

Should a fault or error occur, failure to observe the last guideline could result in scanning of only part of the program, and thus servicing of only part of the process.

Startup procedures

The procedures for COM 265-supported startup of expanded IP 265s are the same as those for single IP 265s.

It is recommended that you proceed as follows:

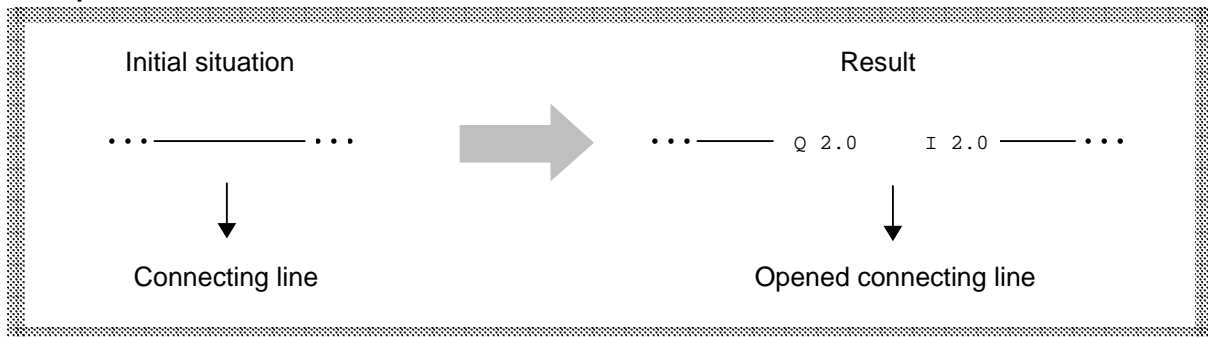
Write the entire IP 265 user program with the COM 265 editor.

Test the complete IP 265 user program off-line.

Divide the IP 265 user program into two programs (program sections) as follows:

- Copy the IP 265 user program
- In each of the two programs, delete the part which is irrelevant to the IP 265 on which that program is to execute and assign each program a different name.
- Assign to the opened connections the addresses of the expansion interface.

Example:



Note

The opened connection addressed with Q 2.0 on the one side in the above diagram must be addressed on the other side with I 2.0.

Compile both programs.

Load each program into the appropriate IP 265, entering the slot number and program name in the "Defaults" form first.

Test both IPs on-line.

When using COM 265, it is possible to test only one IP 265 at a time within the process as a whole. The slot number entry in the "Defaults" form specifies which IP is to be tested.

Simultaneous on-line testing of both IPs can be done using the STEP 5 package's "FORCE VAR" and "STAT VAR" functions.

12.2 Interconnecting Two IP 265s

Siemens provides a prefabricated, shielded standard cable for the electrical connection between two IP 265s (expansion cable).

Insert the two ends of the expansion cable into the 15-pin sub D "Interface" sockets on the two IPs. This establishes the expansion interface connection(Section 2.4).

Note

The expansion inputs/outputs must be operated only with the special expansion cable provided for that purpose (Order No.: See Appendix G).

12.3 Power Supply for Expanded IP 265s

The load voltage (24 V DC) is fed in over the two-pin connector on the frontplate of each IP 265.

Figure 12-2 shows how to connect the expanded IPs to the PLC.

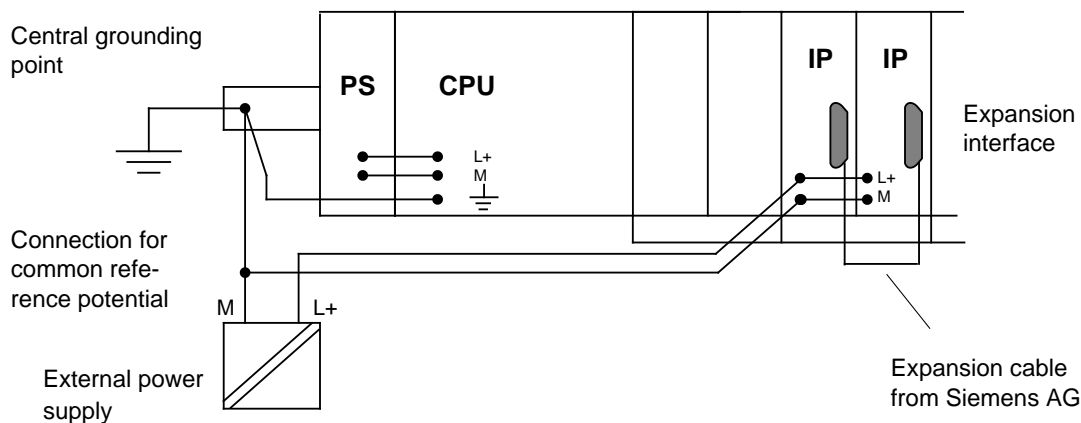


Figure 12-2. Simplified Diagram for Non-Isolated Connection of an Expanded IP 265 to the PLC

Note

When using IP 265s, you must join the IP 265 load voltage connector's zero potential with the CPU's zero potential via an external connection at the central grounding point (Figure 12-2). Care must be taken that a low-resistance connecting cable be used (and that it be as short as possible).

12.4 Addressing the Expansion Inputs/Outputs

In the IP 265 user program, the expansion interface is referenced via fixed addresses.

The expansion interface comprises 8 pins on the IP 265's 15-pin sub D "INTERFACE" socket. These pins may be used as expansion inputs or expansion outputs.

Table 12-1. Addresses of the Expansion Inputs/Outputs

Inputs/outputs	Addresses
Expansion inputs*	I 2.y; y (0 to 7)
Expansion outputs*	Q 2.y; y (0 to 7)

* Alternatives only

Note

You may use each of the 8 expansion interface pins as either expansion input or expansion output (you can program the pins with COM 265).

12.5 Sample Program

The following sample program shows an expansion which became necessary because there was **not a sufficient** number of **digital inputs on one IP 265** to solve the problem at hand. In addition, parameters are interchanged between one of the two IPs and the CPU.

Sample application: Fast shutdown in the event of a malfunction

Problem definition

A web of paper is transported over several driving and guide pulleys for subsequent processing. Four driving pulleys must be immediately shut down if the paper should tear, as a paper jam would otherwise ensue.

Process

Malfunctions are detected as follows:

BEROs are used at ten different points between driving pulleys and guide pulleys to scan the paper web. The 10 BEROs are connected to the IPs' 24 V inputs.

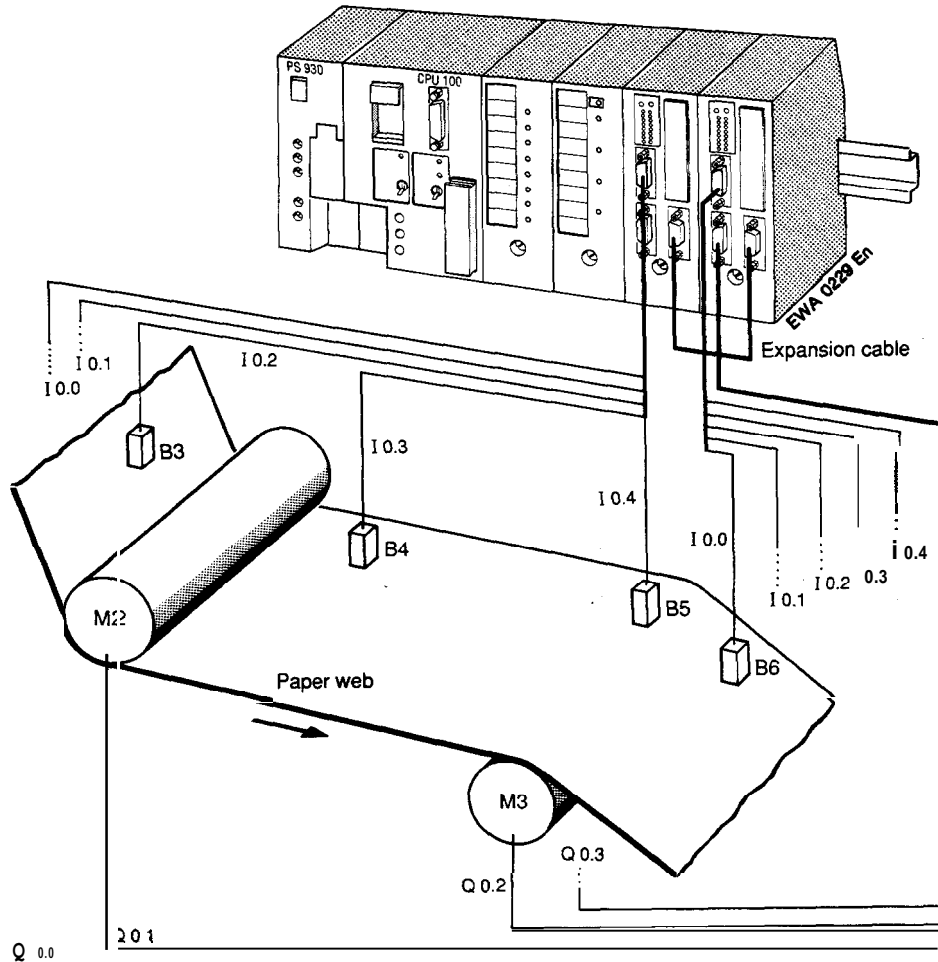


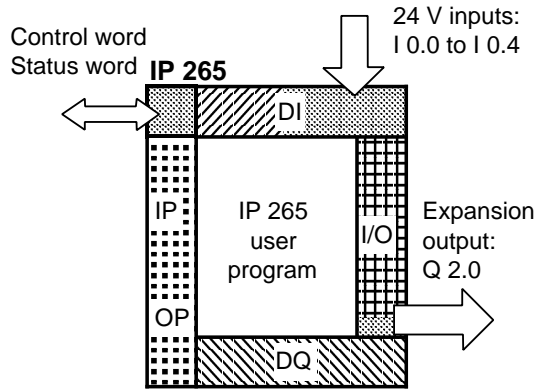
Figure 12-3. Process: Fast Shutdown in the Event of a Malfunction

The drives are controlled as follows:

The CPU forwards the Enable signal for the drives to an IP 265 (slot 3) in the form of a bit parameter. The IP 265 user program logically combines the Enable signal with the process signals from BEROs 6 to 10 or with the other 1P's RLO (slot 2), and immediately shuts down the four drive motors via four digital outputs in the event of a malfunction. The CPU is informed of the shutdown via an output bit parameter. Forwarding of the shutdown info is staggered in time, as data is transferred cyclically between IP and CPU.

Load on the IP 265 interfaces

IP 265 in slot 2:



IP 265 in slot 3:

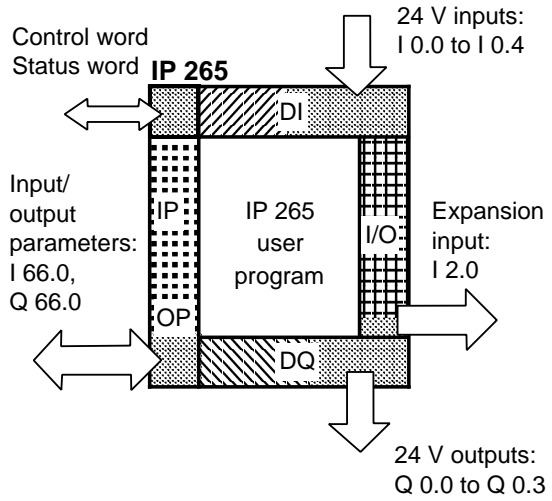


Figure 10-4. Interfaces Used on the Two IP 265s in the Sample Application: "Fast Shutdown in the Event of a Malfunction"

Programming the two IP 265s

- Entries in the local IP 265 assignment list (IP 265 in slot 2):

Operand	Symbol	Comments
I 0.0	B1	DI BERO 1
I 0.1	B2	DI BERO 2
I 0.2	B3	DI BERO 3
I 0.3	B4	DI BERO 4
I 0.4	B5	DI BERO 5
Q 2.0	EXP.OUTPUT	Scan result for BERO 1 to 5

- IP 265 user program: ABFR1 Segment: 1 of 1 (IP 265 in slot 2)

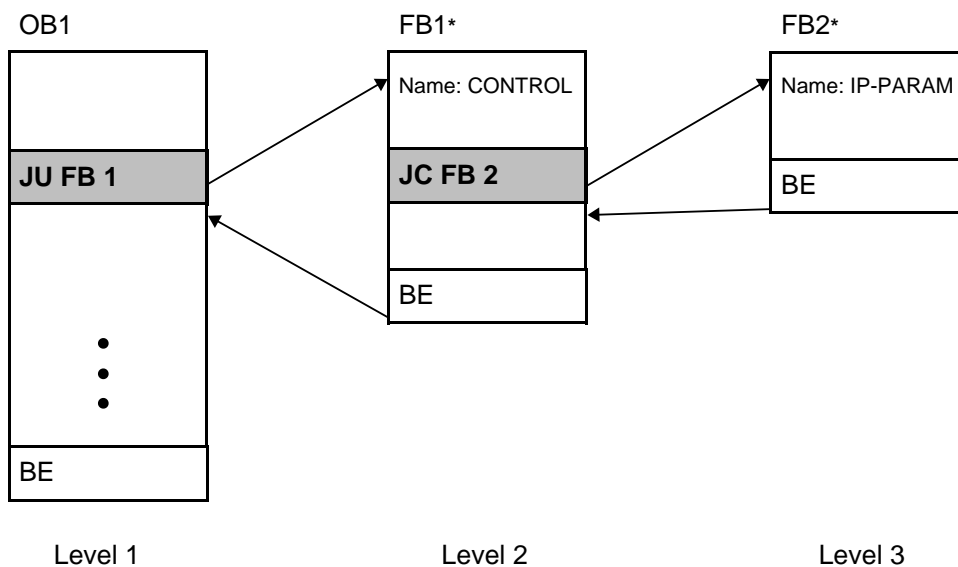


Programming the CPU

Because this sample application includes bilateral data transfers between IP 265 and CPU and a connection via the expansion interface, the CPU user program must be written to include the following:

- The control word for both IP 265s (including matching of their operational states)
 - Evaluation of the status word for both IP 265s and
 - Entry and deletion of parameters
- Structure of the CPU program section for control of two expanded IP 265s:

This section of the CPU program comprises three nesting levels. The function blocks invoked by OB1 are processed cyclically.



- * FB1 "CONTROL": Read and evaluate status word, write control word, invoke FB2 "IP-PARAM"
 FB2 "IP-PARAM": Read and process output parameter, generate input parameters

Figure 10-5. Structure of the CPU Program Section for Controlling two Expanded IP 265s

If the two IPs are in slots 2 and 3, the CPU program section for controlling the IP 265s might be as follows:

- Entries in the SIMATIC assignment list:

Operand	Symbol	Comments
I 90.0		Motor ON/OFF
Q 90.0		Enable signal for motor

- STL in CPU user program:

```

OB 21/22

Segment 1      0000
0001          :L   KH  0000          Scratch flag
0002          :T   FY  64
0003          :L   KH  0200          Set RUN bit for both IP 265s in
0004          :T   QW  80            CPU restart routine
0005          :L   KH  0200
0006          :T   QW  88
.
.
.
ABCD          :BE

OB 1

Segment 1      0000
0000          :JU  FB 1            Call FB 1:
0001 NAME     :CONTROL            Forward control word and
0002          :                                evaluate status word
.
.
.
ABCD          :BE

FB 1

Segment 1      0000          IP 265s in slots 2 and 3
Name :CONTROL

000A          :ANF  M  64.3          Wait cycle expired?
000B          :JC   =M001
000C          :A   I  80.2          If one of the two IP 265s is
000D          :O   I  88.2          under COM 265 control:
000E          :BEC
0010          :A   I  80.0          If one of the two IP 265s is at
0011          :O   I  88.0          STOP:
0012          :S   Q  80.0          --> Set other IP 265 to
0013          :S   Q  88.0          STOP and
0014          :R   Q  80.1          reset the RUN bit in the control
0015          :R   Q  88.1          word for both IP 265s
0016          :BEC

```

(FB 1 continued)

```
0017      :JU   FB  2          --> Initialize and read parameters
0018 NAME  :IP-PARAM          for IP 265 in slot 3
0019      :BEU
001A M001 :L    FY  64          Wait cycles: CPU in restart routine
001B      :L    KF  +1
001C      :+F
001D      :T    FY  64
001E      :BE
```

FB 2

```
Segment 1      0000          IP 265 in slot 3
Name :IP-PARAM

0007      :
0008      :L    IW  90          Load output parameter and
0009      :T    FW  10          store in flag word 10
0007      :
.
.
.
000E      :
000F      :L    FW  12          Transfer flag word 12
0010      :T    QW  90          to input parameter
0011      :BE
```

13	Sample Applications and Programming Aids	
13.1	Sample Programs	13 - 1
13.1.1	Simple Sample Program	13 - 2
13.1.2	Sample Program with Parameter Interchange	13 - 7
13.1.3	Sample Program with Expansion and Parameter Interchange	13 - 13
13.2	Programming Aids	13 - 14

Figures

13-1	IP 265 Interfaces to the CPU and Process I/Os	13 - 1
13-2	Process: Dynamic Measuring System for Profiles	13 - 2
13-3	IP 265 Interfaces Used in the Sample Application: "Length Measuring System for Profiles"	13 - 3
13-4	Process: Synchronizing a Paper Feeder	13 - 7
13-5	IP 265 Interfaces Used in the Sample Application: "Synchronizing a Paper Feeder"	13 - 8
13-6	Structure of the CPU Program Section for IP 265 Control with Parameter Interchange	13 - 11
13-7	Sensor Signal Evaluation	13 - 14

13 Sample Applications and Programming Aids

13.1 Sample Programs

The following sections deal with the programming of the IP 265 and the CPU for control of high-speed subprocesses within a process by presenting practice-related examples.

There are three sample applications, each more complex than its predecessor as regards the user programs and the use of the IP 265 interfaces. You will find one example each for IP 265 control:

- without parameter interchange between IP 265 and CPU (Section 13.1.1)
- with parameter interchange between IP 265 and CPU (Section 13.1.2) and
- with expansion and parameter interchange between IP 265 and CPU (Section 13.1.3).

Different IP 265 interfaces are used in the three examples, depending on the application:

- uses the 24 V inputs and 24 V outputs
- uses the 24 V inputs, 24 V outputs and output parameters
- uses the 24 V inputs, 24 V outputs, input parameters, output parameters and expansion inputs/outputs

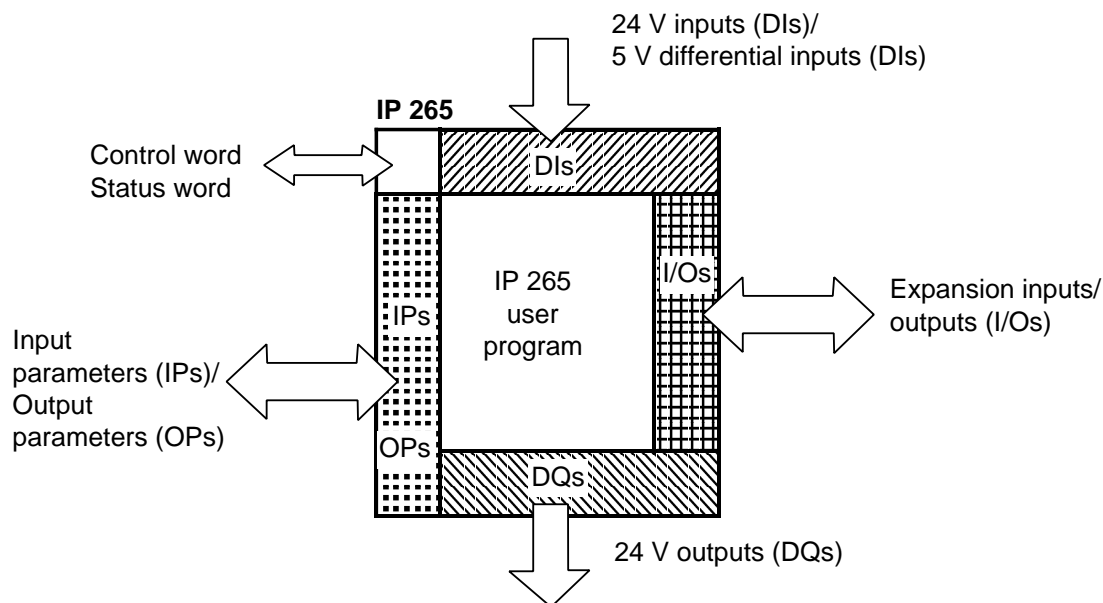


Figure 13-1. IP 265 Interfaces to the CPU and Process I/Os

In the various sections, you will find for each sample program information:

- on the problem definition for the sample application
- on subdividing the process between IP 265 and CPU
- on program conversions in the IP 265 user program and
- on program conversions in the CPU user program.

13.1.1 Simple Sample Program

Sample application for case a):

Controlling a subprocess with an IP 265 without parameter interchange between IP 265 and CPU.

A high-speed subprocess is to be autonomously controlled by the IP 265. Communication between CPU and IP is limited to forwarding of the RUN command for the IP.

Sample application: Dynamic length measuring system for profiles

Problem definition

Length profiles are manufactured on an assembly line. The profiles are transported on a conveyor belt at a speed of 20 m/min. The profile length is measured while the conveyor belt is in motion. The length tolerance is 0.1 mm.

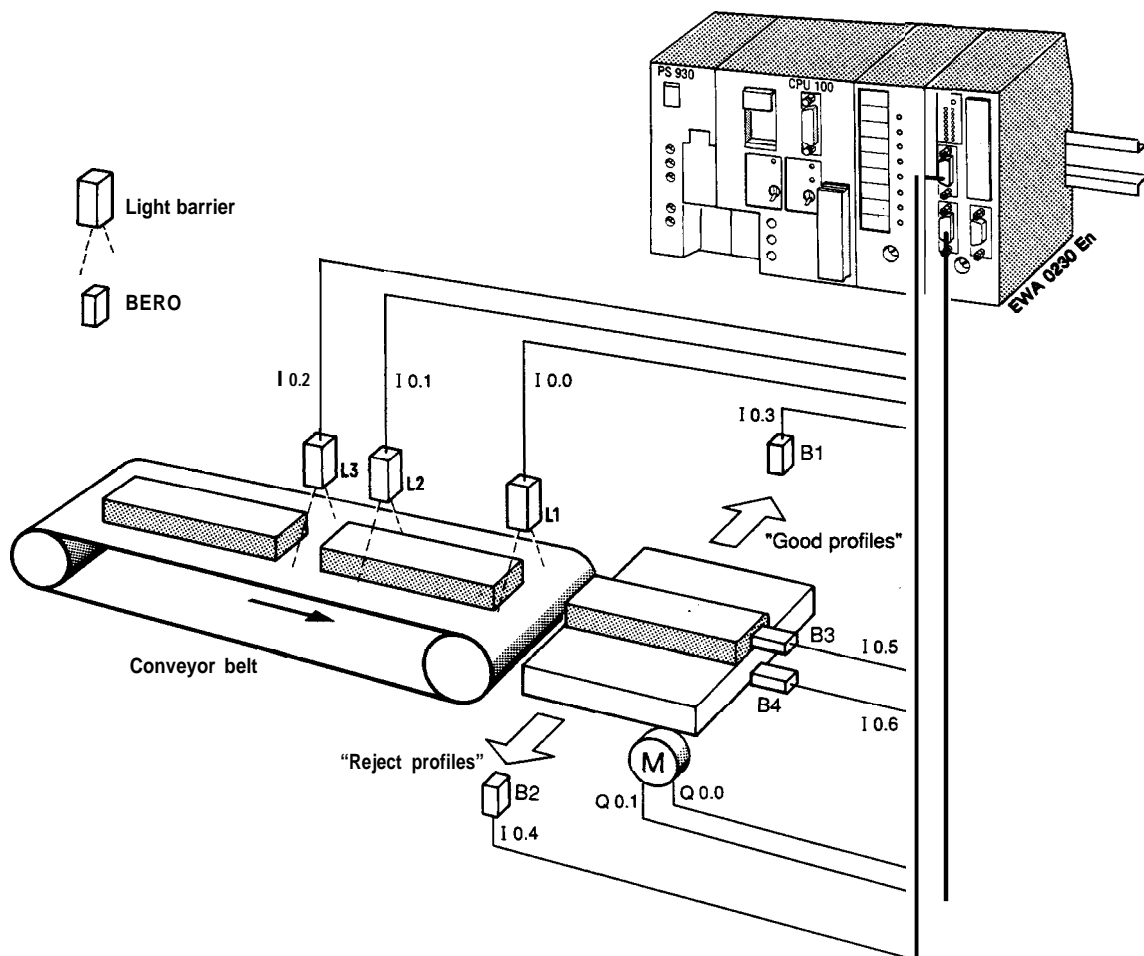


Figure 13-2. Process: Dynamic Measuring System for Profiles

Process

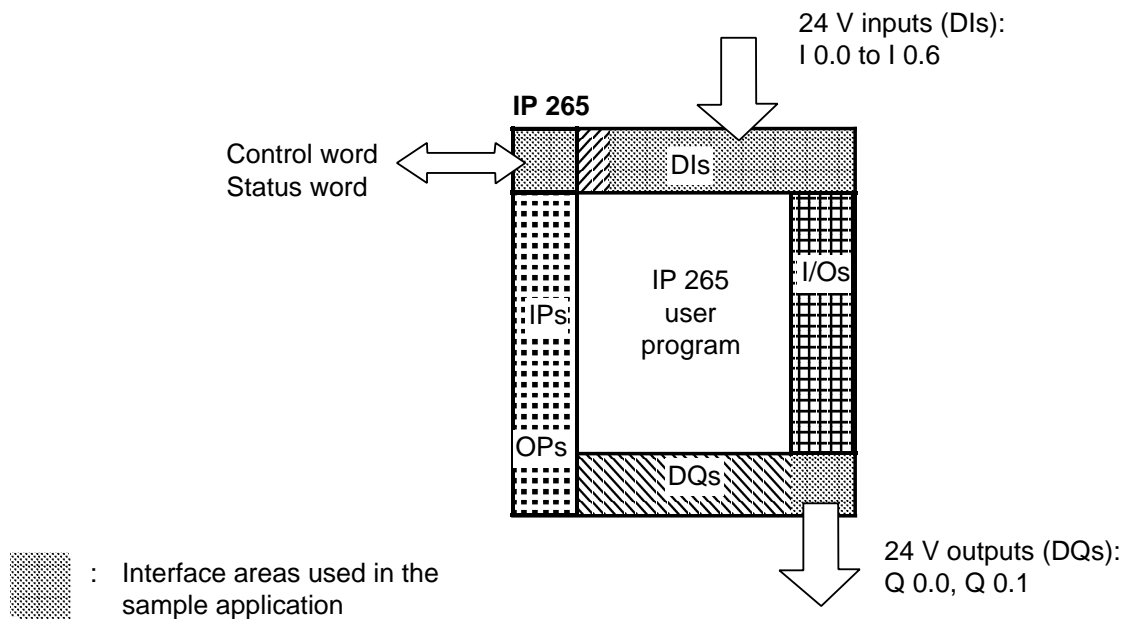
Profile length control procedures are as follows:

- Light barriers L2 and L3 are 0.1 mm apart (permissible length tolerance). The mean distance between L1 and L2/L3 represents the normal length of a profile. Light barriers L1, L2 and L3 are connected to the IP 265's 24 V inputs.
- When a profile triggers a positive edge at light barrier L1 while in transport, the signals from light barriers L2 and L3 are evaluated simultaneously. If L2 generates no signal, the profile is too short; if L3 generates a signal, the profile is too long. In both cases, the profile must be rejected.

Profiles are rejected as follows:

- The profiles are pushed from the conveyor belt onto a distributing slide which separates the "good" profiles from the "reject" profiles. The IP 265 has full control of the distributing slide.
- BEROs B1 and B2 represent the right and left final position of the distributing slide. BEROs B3 and B4 indicate whether the profile and distributing slide are in the proper position so that distribution can begin. The distributing slide is driven by a motor which is switched to ON/OFF or clockwise/counterclockwise rotation via two IP 265 outputs.

Load on the IP 265 interfaces



**Figure 11-3. IP 265 Interfaces Used in the Sample Application:
"Length Measuring System for Profiles"**

Programming the IP 265

- Entries in the local IP 265 assignment list (IP 265 in slot 2)

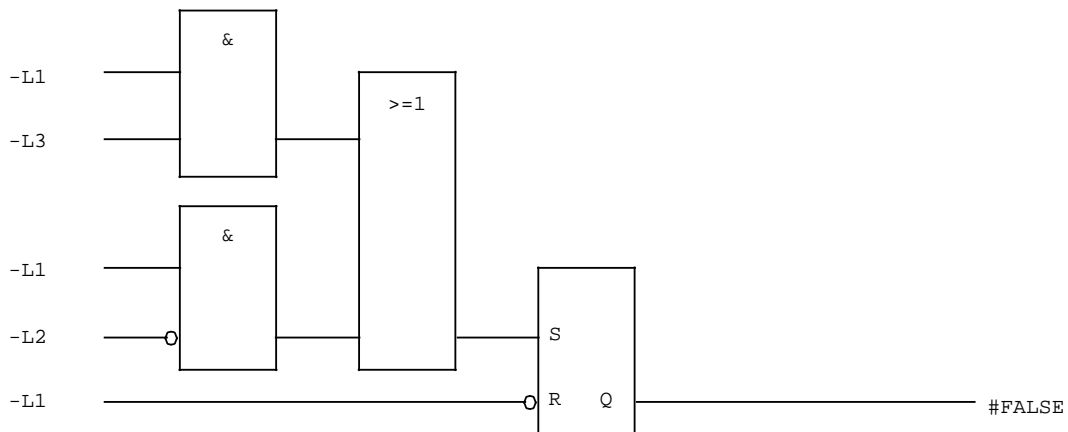
Operand	Symbol	Comments
I 0.0	L1	DI Light barrier 1: Measuring release
I 0.1	L2	DI Light barrier 2: Measuring signal
I 0.2	L3	DI Light barrier 3: Measuring signal
I 0.3	B1	DI BERO 1: Final pos. of distributing slide
I 0.4	B2	DI BERO 2: Final pos. of distributing slide
I 0.5	B3	DI BERO 3: Profile on belt
I 0.6	B4	DI BERO 4: Slide at home position
Q 0.0	ON/OFF	DQ Motor ON/OFF
Q 0.1	CLKW/CCLKW	DQ Motor clockwise/counterclockwise
#	FALSE	Scan: Faulty profile

- CSF for the IP 265 user program:

IP 265 user program: PROF1

Segment: 1 of 3

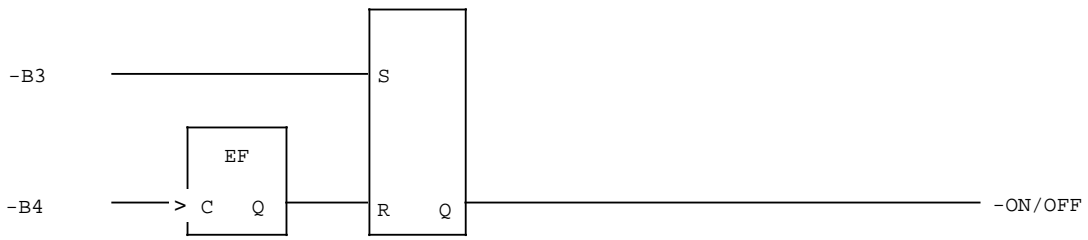
(faulty profile)



IP 265 user program: PROF1

Segment: 2 of 3

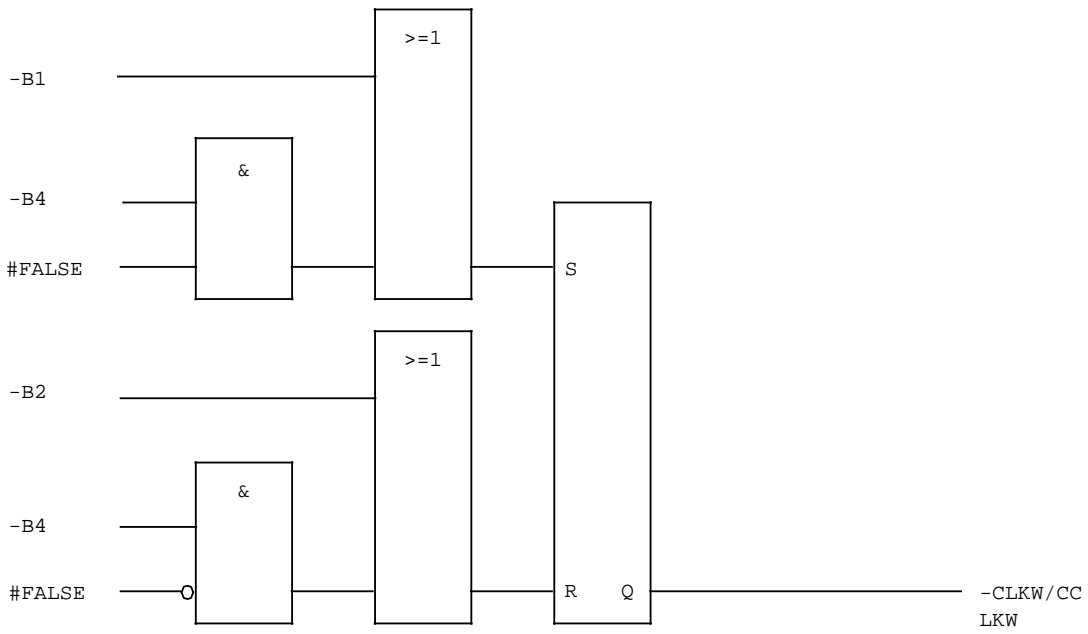
(ON/OFF control)



IP 265 user program: PROF1

Segment: 3 of 3

(motor rotation)



Programming the CPU

As there is no parameter interchange between IP 265 and CPU in this example, the CPU need only set the RUN bit in the control word.

The RUN command for the IP 265 is forwarded to the IP 265 in the CPU's restart routine.

If the IP is in slot 2, the section of the CPU user program for IP 265 control might look as follows:

- STL in the CPU user program:

OB 21/22

Segment 1	0000		
0000	:L	KH 0200	Set RUN bit for IP 265 in
0001	:T	QW 80	the CPU restart routine
.			Normal CPU user program:
.			No processing of IP 265 input
.			or output data
ABCD	:BE		

13.1.2 Sample Program with Parameter Interchange

Sample application for case b):

IP 265 control of a subprocess with parameter interchange between IP 265 and CPU.

If data (such as preset values for timers, counters or simple bit information) is to be interchanged between IP 265 and CPU, it must be linked into the IP 265 user program as input parameters (CPU to IP) or output parameters (IP to CPU).

Sample application: Synchronizing a paper feeder

Problem definition

On an assembly line, a strip of paper and a strip of cardboard are transported next to one another on a conveyor belt in a continuous process. Prerequisite for high-quality postprocessing is the synchronism of the two strips (the front edges of the strips must always be exactly aligned). Even minor deviations require a transport adjustment.

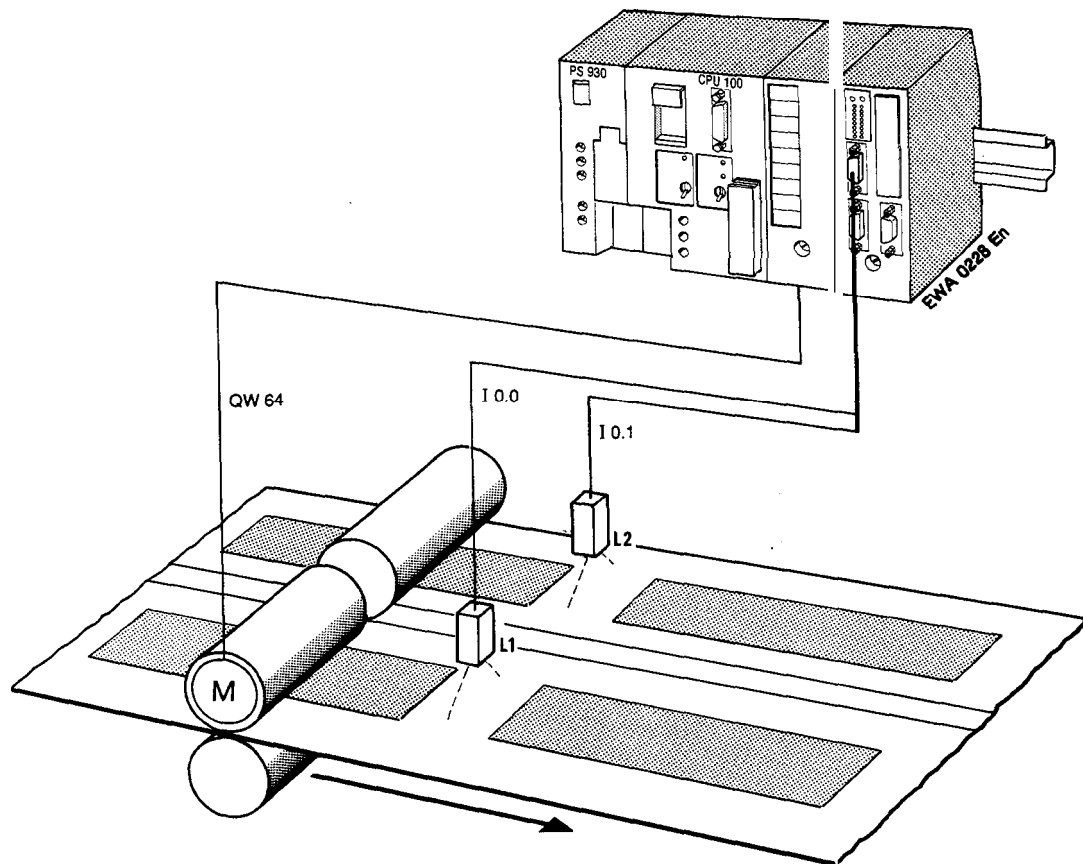


Figure 13-4. Process: Synchronizing a Paper Feeder

Process

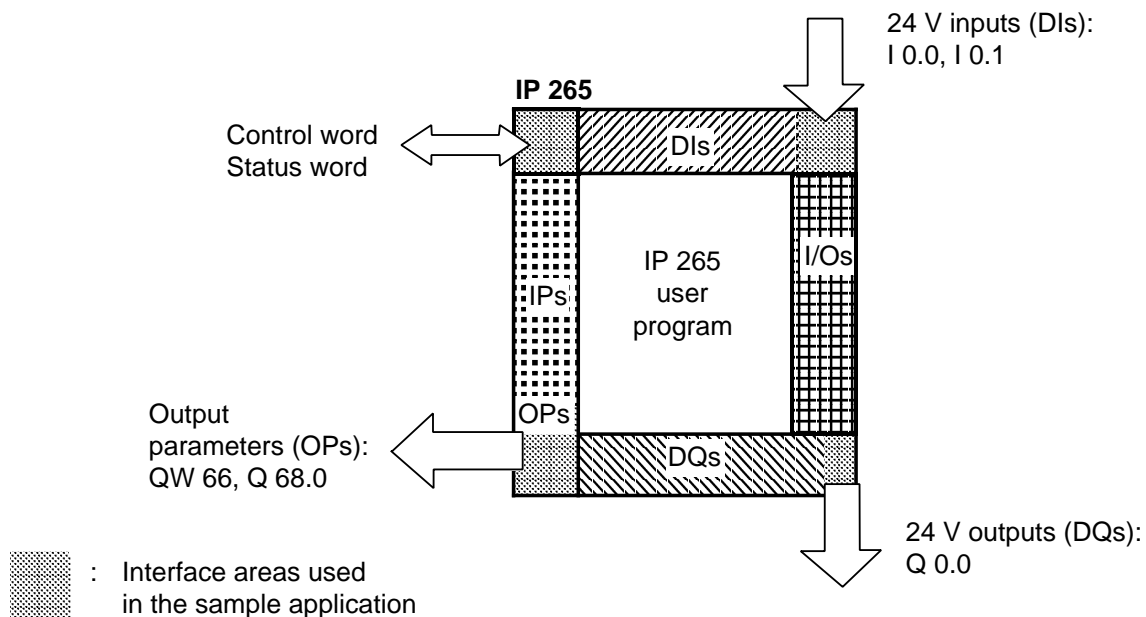
Synchronous operation is controlled as follows:

- Two light barriers (L1 and L2) are arranged so that they are side by side and at a right angle to the conveyor belt's transport direction. The light barriers are connected to the IP 265's 24 V inputs, and implement "GATE" functionality for an up/down counter in the IP 265.

The paper or cardboard strip is adjusted as follows:

- A counter is loaded for each new measurement with a count value (CV=100). If the front edge of a strip triggers one of the light barriers, a clock-pulse generator generates a 64 kHz frequency at the counting input (whether at the CU or the CD input depends on which barrier was triggered). Counting is halted when the second light barrier is triggered.
- The IP 265 forwards the count to the CPU as output parameter, and the CPU user program postprocesses this count to adjust the transport speed of the relevant strip. The number of pulses counted is one means of ascertaining the amount of deviation. A comparison with the count shows which strip was too fast or too slow.
- The CPU user program drives an auxiliary motor via an analog output module to adjust the transport speed.

Load on the IP 265 interfaces



**Figure 13-5. IP 265 Interfaces Used in the Sample Application:
"Synchronizing a Paper Feeder"**

Programming the IP 265

- Entries in the local IP 265 assignment list (IP 265 in slot 2):

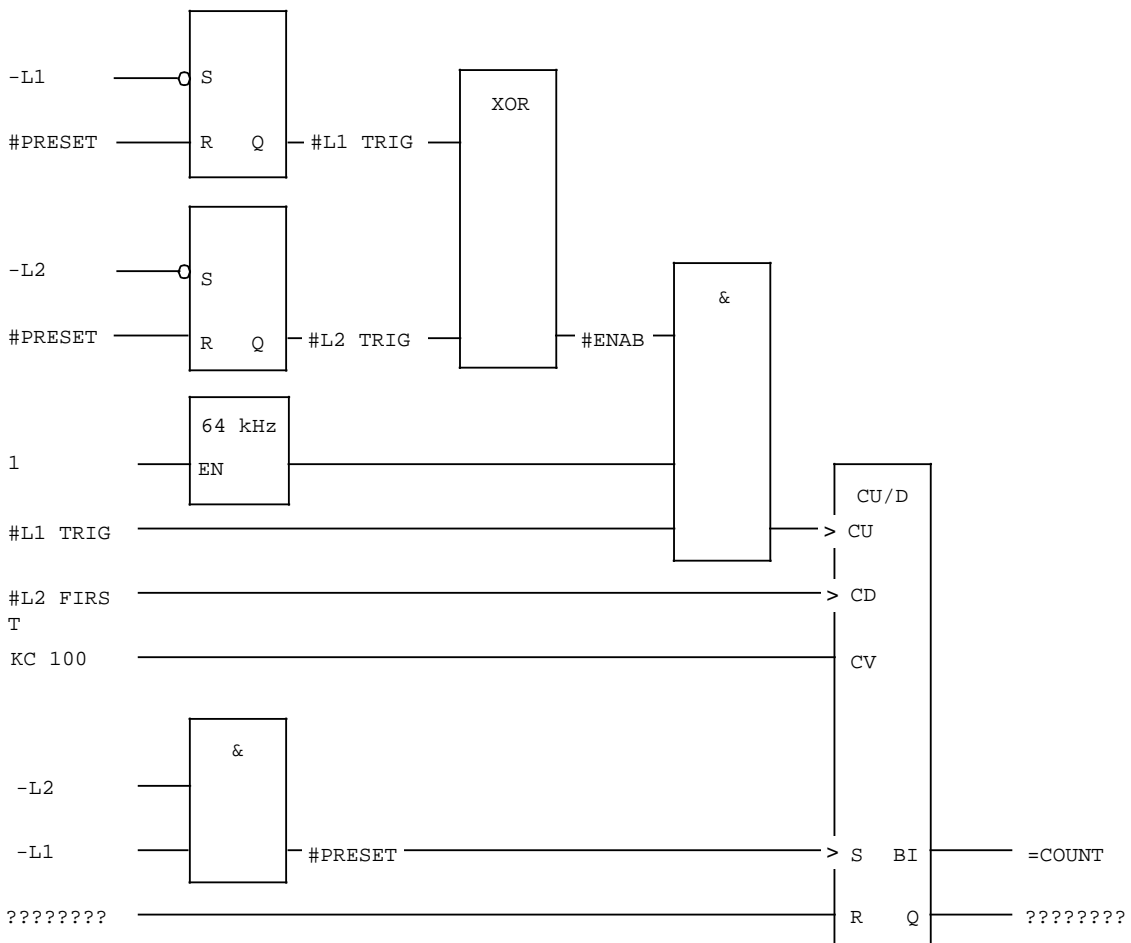
Operand	Symbol	Comments
I 0.0	L1	DI Light barrier 1: Measuring signal
I 0.1	L2	DI Light barrier 2: Measuring signal
QW 66	COUNT	Count for correction
Q 68.0	VALID	Scan: Counting terminated
KC 100		Preset value for counter
#	PRESET	Load preset value 100 into counter
#	L1 TRIG	Light barrier 1 triggered
#	L2 TRIG	Light barrier 2 triggered
#	ENAB	Enable for counting pulse
#	L2 FIRST	Light barrier 2 counts

- CSF for the IP 265 user program:

IP 265 user program: PAPPE

Segment: 1 of 3

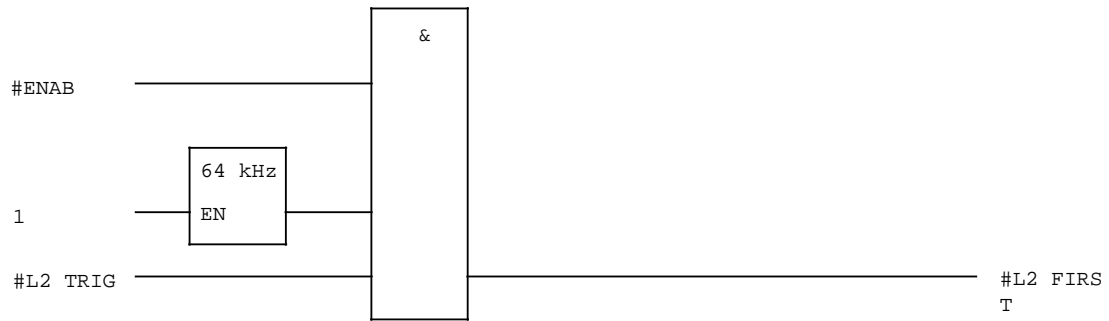
(Count deviation)



IP 265 user program: PAPPE

Segment: 2 of 3

(Count L2)



IP 265 user program: PAPPE

Segment: 3 of 3

(Count valid ?)

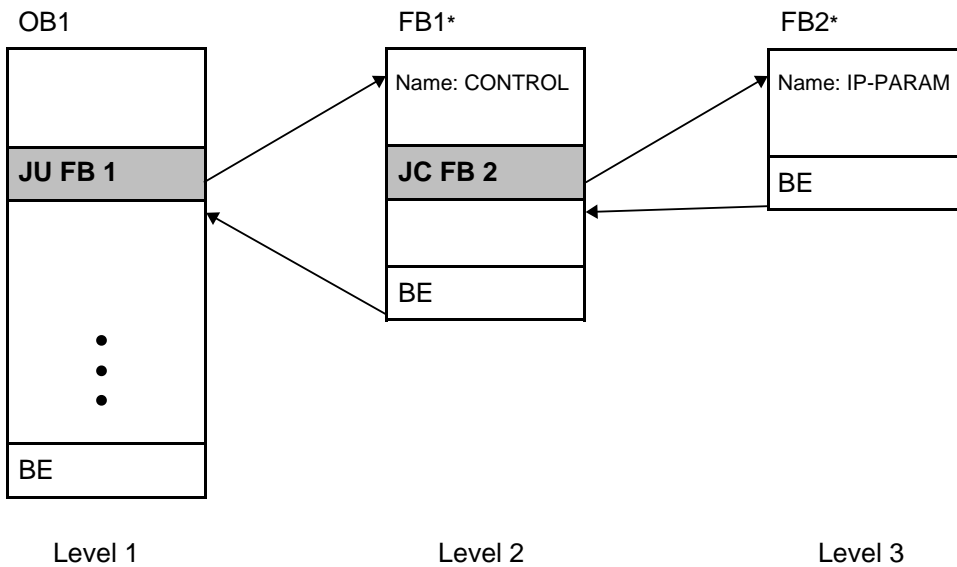


Programming the CPU

Because the CPU and the IP 265 interchange parameters in this example, the CPU user program must be written to include not only the control word (IP 265 RUN, IP 265 STOP) and the status word (operational status, error diagnostics), but also reading of the output parameters.

- Structure of the CPU program section for controlling the IP 265

This section of the CPU program comprises three nesting levels. The function blocks invoked by OB1 are processed cyclically.



* FB1 "CONTROL": Read and evaluate status word, call FB2 "IP-PARAM"
 FB2 "IP-PARAM": Read and process output parameters

Figure 13-6. Structure of the CPU Program Section for IP 265 Control with Parameter Interchange

If the IP 265 is in slot 2, the section of the CPU user program for controlling the IP 265 might look as follows:

- Entries in the SIMATIC assignment list:

Operand	Symbol	Comments
IW 82		Count for correction
I 84.0		Scan: Counting terminated

- STL in the CPU user program:

OB 21/22

```

Segment 1      0000
0000          :L   KH  0200          Set IP 265's RUN bit in the
0001          :T   QW  80           CPU restart routine
0002          :
.
.
.
ABCD          :BE

```

Normal CPU user program:
No processing of IP 265 input
or output data

OB 1

```

Segment 1      0000
0000          :JU  FB 1           Call FB 1:
0001 NAME     :CONTROL          Define control word and evaluate
0002          :                 status word
.
.
.
ABCD          :BE

```

Normal CPU user program:
No processing of IP 265 input
or output data

FB 1

```

Segment 1      0000          IP 265 in slot 2
Name :CONTROL

000A          :A   I   80.1       IP 265 in RUN mode and
000B          :AN  I   80.2       IP 265 not under COM 265 control:
000C          :JC  FB 2          --> Initialize and process
000D NAME     :IP-PARAM          IP 265 parameters
000E          :BE

```

FB 2

```

Segment 1      0000          IP 265 in slot 2
Name :IP-PARAM

0007          :
0008          :L   IW  82          Load output parameters and store
0009          :T   FW  10         in flag word 10
0008          :L   IB  84          Load output parameter and store
0009          :T   FY  12         in flag byte 12
0007          :
.
.
.
ABCD          :BE

```

Processing of output parameters
by the CPU user program.

13.1.3 Sample Program with Expansion and Parameter Interchange

Sample application for case c):

Controlling a subprocess with expansion and parameter interchange between IP 265 and CPU.

If an IP 265 user program cannot be implemented on a single IP 265 because of its size or because of the number of digital inputs/outputs needed, the IP 265 can be expanded to include a second IP.

The IP 265 user program is then divided into two programs, one for each IP. The two programs are then rejoined via the expansion interface.

The sample program shows an expansion which became necessary because the number of digital inputs on one IP 265 was insufficient. In addition to expansion, the two IPs interchange parameters with the CPU.

Sample application: Fast shutdown in the event of a malfunction

The sample application is described in detail in Section 12, "IP 265 Expansion".

13.2 Programming Aids

This section discusses a number of programming aids for the IP 265 user program.

Programming aids (also called software tools) are program fragments containing complex functions which must often be included in control programs for high-speed processes and which can be easily linked into the IP 265 user program.

You will find programming aids for the following examples:

- Pulse evaluator for position decoder
 - 1-fold pulse evaluation
 - 2-fold pulse evaluation
- High-speed, reproducible resetting of an output

Example: Pulse evaluator for position decoder

An incremental position decoder supplies two in-quadrature pulse signals displaced by 90°. The frequency of the pulse signal may not exceed 29 kHz.

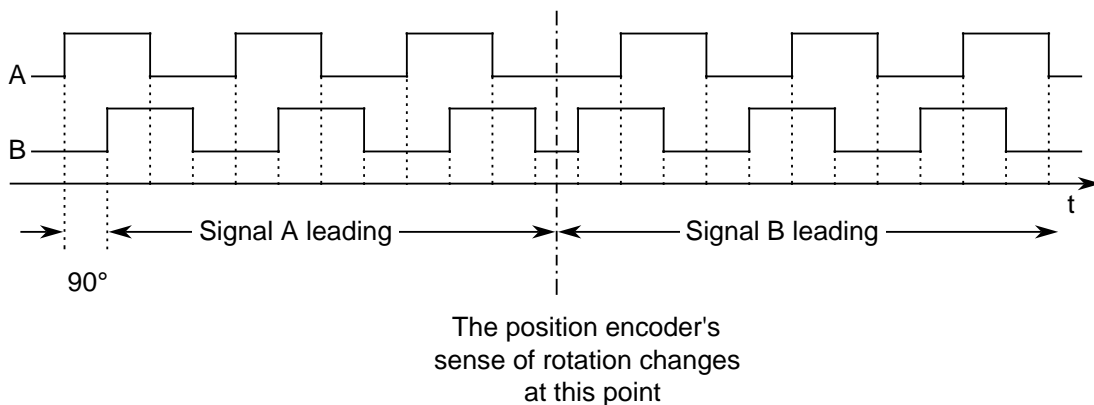
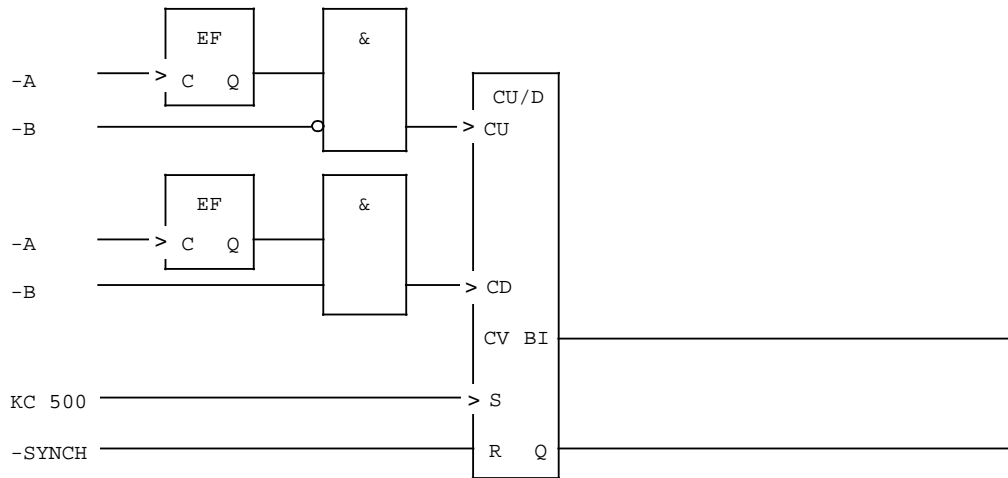


Figure 13-7. Sensor Signal Evaluation

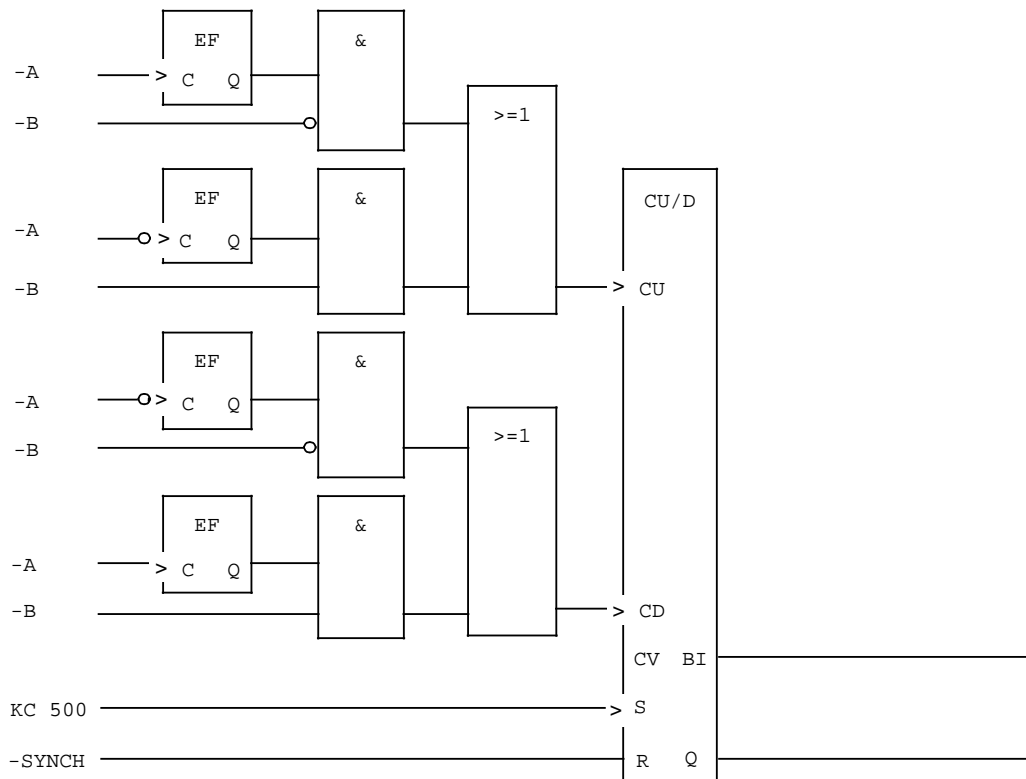
The reference point is determined by a SYNCH signal at the counter's Reset input.

Signals A, B and SYNCH can be routed to both the 24 V inputs and the 5 V differential inputs (RS422).

- 1-fold pulse evaluation:



- 2-fold pulse evaluation:



14 Standard Function "Counter"

((Space holder for the documentation for the standard function counter))

What is the standard function counter?

This standard function is an IP 265 user program which was written and pretested by the Siemens AG. In automation technology, the counting of high-speed processes is a field in itself. For this reason, Siemens has developed a memory submodule containing a standard function called "Counter".

Functional overview

The counter function can be used together with the IP 265 in the S5-90U, S5-95U and S5-100U programmable controllers and in the ET 200 system (beginning revision level 2 of the IM 318-B). The counters function completely determines the IP 265's functionality. All interfaces to the standard program are IP 265 interfaces to the SIMATIC environment or to the process I/Os.

The standard program with the name "Counter" provides three different counting functions. These can be subdivided in turn into two categories:

- Two 16-bit counters
The two 16-bit counters are independent of one another, and each can be operated as up counter (normal counting function) or down counter (periodic counting function).
- One 32-bit counter
For counting functions which require considerably more scope, the two 16-bit counters can be combined into one 32-bit counter (cascade function). The 32-bit counter is a down counter (periodic counting function).

On the STEP 5 end, you can forward setpoints and control signals and read actual values and diagnostic signals over the external I/O bus.

Standard package

The counters program must be ordered separately, and is supplied on an EPROM memory submodule.

Note

The counters program has its own separate documentation. This documentation is sent with the program, and is not included in this manual. We have, however, reserved space in this manual for the standard function counter, and would recommend that you insert the documentation for the counter program into the manual in the space provided.

Order Numbers

See Appendix G: "Accessories and Order Numbers"

Appendices

Appendix A	...	Diagnostics and Error Messages
Appendix B	...	Technical Specifications
Appendix C	...	Dimension Drawing of the IP 265
Appendix D	...	Keyboard Layout for COM 265 Editing Functions on the Programmer
Appendix E	...	Glossary
Appendix F	...	Active and Passive Faults in Automation Systems
Appendix G	...	Accessories and Order Numbers

A Diagnostics and Error Messages		
A.1	LEDs	A - 1
A.2	IP 265 Error Messages	A - 2

A Diagnostics and Error Messages

A.1 LEDs

The STOP and RUN LEDs show the following module modes:

Operating Status of the CPU	LEDs on the IP 265		Diagnostics	Operating Status of the IP 265
	STOP LED	RUN LED		
RUN	Steady light	-	Inputs/outputs passive	STOP
RUN	Flashing light	-	Group fault Inputs/outputs passive	STOP/ Hard STOP
RUN	Flashing light	Steady light	Short circuit at output (24 V) Inputs/outputs active	RUN
RUN	-	Steady light	Inputs/outputs active	RUN
RUN	High speed flashing	-	Loading over I/O bus or loading from memory submodule	STOP
STOP	Steady light	-	Inputs/outputs passive *	STOP
STOP	Flashing light	-	Group fault Inputs/outputs passive	STOP/ Hard STOP
STOP	Flashing light	Steady light	Short circuit at output (24 V) LED goes on only when inputs/outputs have been configured as active *	RUN
STOP	-	Steady light	Inputs/outputs active *	RUN
STOP	High-speed flashing	-	Loading from memory submodule	STOP

* Can be configured using COM 265

Note

The expansion outputs cannot be made passive.

A.2 IP 265 Error Messages

The IP 265 provides sophisticated error/fault evaluation options. The IP 265 displays operator errors and hardware faults:

- Group faults are flagged in the status word
- The appropriate error bit is set in the status word
- The IP 265 goes to STOP or hard STOP
- The STOP LED flashes

Error/Fault	Operating Status of the IP 265	Error bit in byte 1 of the status word							
		7	6	5	4	3	2	1	0
Invalid memory submodule (not an S5-90 submodule)	Hard STOP	Bit for invalid memory submodule:							
		X	X	X	X	X	0	0	1
EPROM memory submodule contains no data or contains invalid IP 265 user program	Hard STOP	Bit for empty EPROM submodule or for submodule containing invalid program data:							
		X	X	X	X	X	0	1	0
EEPROM memory submodule contains invalid IP 265 user program	STOP	Bit for EEPROM containing invalid program data:							
		X	X	X	X	X	1	0	1
Transfer error during loading	STOP	Load error bit:							
		X	1	X	X	X	X	X	X
Invalid command	STOP	Command error bit:							
		1	X	X	X	X	X	X	X
Invalid input parameter (parameter out of range)	STOP	Command error bit:							
		1	X	X	X	X	X	X	X
FPGA defective or module fault	Hard STOP	Module fault bit:							
		X	X	X	1	X	X	X	X
Memory submodule inserted or removed while live (POWER ON)	Hard STOP	Bit for impermissible insertion or removal of memory submodule:							
		X	X	X	X	X	1	1	1

(Continued)

Error/Fault	Operating Status of the IP 265	Error bit in byte 1 of the status word							
		7	6	5	4	3	2	1	0
Wirebreak/No 24 V voltage	STOP	I/O error bit:							
		X	X	1	X	X	X	X	X
Short circuit in outputs (I/O not ready)	RUN	I/O error bit:							
		X	X	1	X	X	X	X	X

Note

A "Hard STOP" can be exited only after a POWER OFF.

Note

A "Short circuit in outputs" fault does not set the IP 265 to STOP. The group fault and I/O error bits are set in the status word, the STOP LED flashes and the RUN LED shows steady light.

B Technical Specifications

B Technical Specifications

Module	
Permissible ambient temperature	
- Horizontal configuration	0 to+60 °C
- Vertical configuration	0 to+40 °C
Current consumption from+9 V (CPU)	<175 mA
Insulation rating	To VDE 0160
Signal status display	For 24 V inputs and 24 V outputs only (green LEDs)
Status indicators for RUN (green LED)	for STOP (red LED)
Memory submodule	EPROM/EEPROM
Address code for ET 200 (Slow mode only)	223
Weight	approx. 300 g
Digital 24 V inputs	9-pin sub D connector
Number of inputs	8
Galvanic isolation	no
Status indication	Yes, on 5 V side
Input voltage L+	
- Nominal value	24 V DC
- for "0" signal	0 to 5 V
- for "1" signal	11 to 30 V (IEC 65 A)
Input current for "1" signal	typ. 6.5 mA (IEC 65 A)
Connection of 2-wire BERO	possible (quiescent current 1.5 mA)
Input frequency	max. 10 kHz
Cable length (shielded)	max. 100 m
Input circuit delay	
- rising edge	typ. 15 µs
- falling edge	typ. 10 µs
5 V differential inputs	15-pin sub D HD socket connector
Number of type of input signals	3 differential signals to RS 422
Input frequency	max. 58 kHz
Pulse length	
- Low level	min. 8.6 µs
- High level	min. 8.6 µs
Cable length (shielded)	max. 32 m

Digital 24 V outputs		9-pin sub D socket	
Number of outputs		8	
Galvanic isolation		no	
Status indicators		yes, on 5 V side	
Short-circuit protection		yes, electronically clocked	
Load voltage L+		24 V DC	
- Nominal value		20 to 30 V	
- permissible range			
Output current at "1" signal		0.5 A at 60 °C	
Permissible total current of the outputs		2 A at 60 °C	
Parallel switching of the outputs		possible in pairs ($I_{Outp} \times 0.8 \times I_{Nom}$)	
Output frequency in the case of resistive load	max.	1 kHz	at 15 mA load*
	max.	2 kHz	at 50 mA load*
	max.	4 kHz	at 500 mA load*
Cable length	max.	100 m	
Lamp load	max.	2 W	
Residual current at "0" signal	max.	1 mA	
Voltage drop with "1" signal	max.	1 V	
Voltage induced on circuit interruption limited to		- 15 V	
Output circuit delay			
- rising edge	typ.	10 µs	
- falling edge			
dependent on resistive load:	typ.	150 µs	at 15 mA load*
	typ.	90 µs	at 50 mA load*
	typ.	70 µs	at 500 mA load*
Expansion inputs and outputs		15-pin sub D HD socket	
Number of inputs and outputs		8 (any mix of I/Os configurable)	
Connector for 24 V load voltage		2-pin	
Permissible cable cross-sections			
- Flexible H07V-K cable with connector sleeve		0.5 to 1.5 mm ²	
- Solid H07V-U cable		0.5 to 2.5 mm ²	

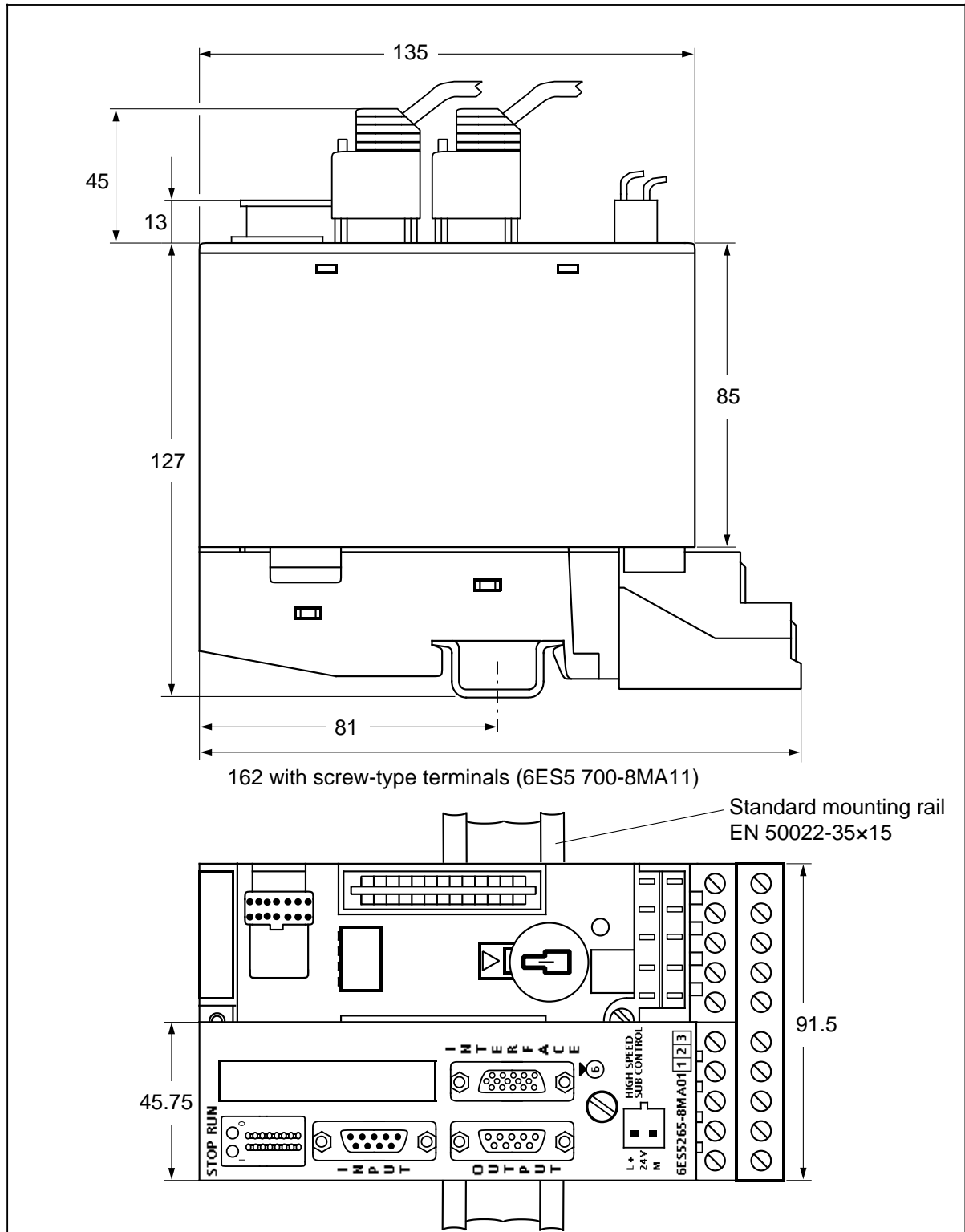
* Peak value (not an r.m.s specification)

Note

The IP 265 can be used in the S5-90U, S5-95U, S5-100U (CPU 100 from 6ES5 100-8MA02, CPU 102 from 6ES5 102-8MA02 or CPU 103) and the ET 200U (IM 318-B from Version 2).

C Dimension Drawing of the IP-265

Dimension Diagram of the IP 265



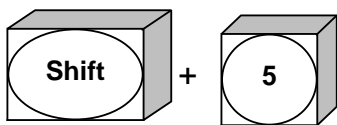
D Keyboard Layout for COM 265 Editing Functions on the Programmer

D Keyboard Layout for COM 265 Editing Functions on the Programmer

The COM 265 editing functions on the programmer are all invoked via the programmer's numeric keypad. Nonetheless, a number of editing functions can also be invoked via other keys or key combinations, as described in Section 8.5.4.

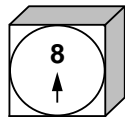
The information on the following pages describes how to initiate all COM 265 editing functions on the programmer.

Inserting comments

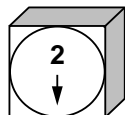


- : - Press <Shift> <5> once only to enter the segment name
- Press <Shift> <5> twice in succession to enter segment comments

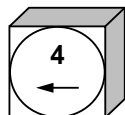
Cursor control



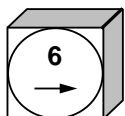
- : Move cursor up one line in an input field;
Move cursor up on a grid-oriented basis in a control system flowchart



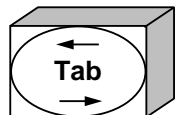
- : Move cursor down one line in an input field;
Move cursor down on a grid-oriented basis in a control system flowchart



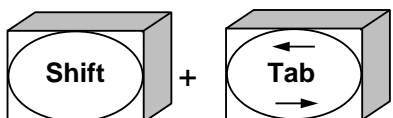
- : Move cursor one character to the left in an input field;
Move cursor to the left on a grid-oriented basis in a control system flowchart



- : Move cursor one character to the right in an input field;
Move cursor to the right on a grid-oriented basis in a control system flowchart



- : Move cursor to the next input field;
Move cursor from input bar to output bar in a control system flowchart



- : Move cursor to the preceding input field;
Move cursor from output bar to input bar in a control system flowchart

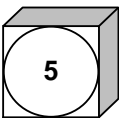
"Move" screen:

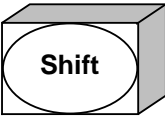
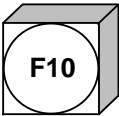
- : Scroll up one line
- : Scroll down one line
- + : Page down
- + : Page up

Segment handling:

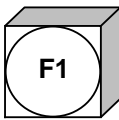
- : Page to preceding segment
- : Page to next segment
- + : Insert a segment in front of the on-screen segment. The segment numbers of all following segments are incremented by one.
- + : Delete the on-screen segment. The numbers of all following segments are decremented by one.
- + : Add a new segment at the end of the program.

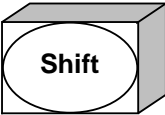
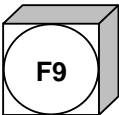
Go from "Simulator" to "Editor" and back:

 : Correction key
Go directly from the simulator to the COM 265 control system flowchart editor without screening the "Function Selection" menu.

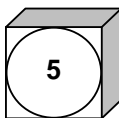
 +  : Return to the simulator. The changes made in the CSF can be stored (user prompt).

Go from "Compiler" to "Editor" and back:

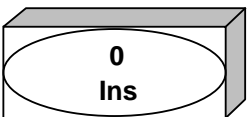
 : Function key <F1> "Editor"
Go directly from the compiler to the COM 265 control system flowchart editor without screening the "Function Selection" menu.

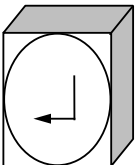
 +  : Return to the compiler. The changes made in the CSF can be stored (user prompt).

Go from "Editor" to "Assignment list" and back:

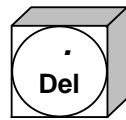
 : Go directly from the control system flowchart editor to the symbolic editor (assignment list line) without exiting the "Editor" form (Section 9.4.3).

Store functions:

 : Store key

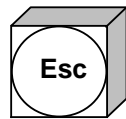
 : Return key

Delete function:

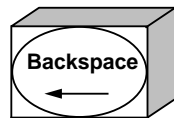


: Delete key
Delete the characters, language elements or inserted inputs of a language element to which the cursor is set.

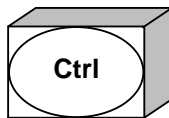
Special functions not invoked on the numeric keypad:



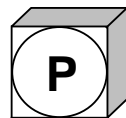
: Abort key (ESC)
Sometimes accompanied by a prompt asking "Are you sure?".



: Information on the current function key menu in the selected COM 265 screen form.



+



: Print the segment currently on the simulator screen in CSF format. When invoked in text format, the entire IP 265 user program is printed out. When invoked in the symbolic editor, a list of all operands is printed out.

Note

Entries in the COM 265 screen form are supported by function keys <F1> to <F10>.

E Glossary

E Glossary

COM 265 language element

An elementary function in the IP 265 user program, such as logic operations, counters, timers, comparators and the like.

Control bit

Informs the CPU application program that the IP 265 is working under COM 265 control and may not be serviced by the CPU.

Control word

Data word via which the CPU application program acts upon the IP 265 user program. Via the control word, the CPU can set the IP 265 to STOP or RUN. To make this possible, the CPU program must set either the RUN bit or the STOP bit in the control word.

CPU application program

Program which executes on an S5-100 CPU and which controls both the process and the IP 265 via commands (STOP/RUN) and parameters.

Debouncing time

Delay required to match the high-speed 24 V inputs to simple sensors. A debouncing time of 1 to 2 ms can be configured with COM 265.

Direct address

Fixed addresses for operands in the IP 265 user program with which the IP 265 interfaces are referenced in the IP 265 user program.

Direct identifier Direct address

FPGA (Field Programmable Gate Array)

Electronic component of the IP 265 whose function is determined by the user with the aid of a programming package. In contrast to microprocessors, an FPGA does not process program sequences, but rather interconnects hardware structures. Components of this type are therefore much faster than standard processors. The IP 265 is equipped with an FPGA whose functionality is determined by the contents of a memory, thus enabling arbitrary reloading.

FPGA clock pulse

Clock pulse with which the FPGA's inputs and outputs are read or written and the retentive COM 265 language elements are timed.

Group error bit

Bit in the IP 265 status word which the IP 265 sets each time an error is detected (e.g. command error or load error). An error evaluation routine can be initiated by scanning the group error bit in the CPU application program.

Hard STOP

An IP 265 STOP state which can be exited only with a POWER OFF. The IP 265 is set to hard STOP mode when it has been equipped with a faulty EPROM submodule, when the submodule is inserted or removed during POWER ON, or when the module is defective.

Identifier Direct, symbolic address

IInput data

Data which the CPU forwards to the IP 265 via the process output image (PIQ). It consists of the IP 265 control word and input parameters for the IP 265 user program, if any.

IInput parameters Parameters**I**IP 265 assignment list

Local operand assignment list, which is independent of the global PLC assignment list. The IP 265 assignment list contains all symbolic and direct addresses of the operands in the IP 265 user program.

IIP 265 load

Program load capacity of the FPGA in the IP 265 in percent (%).

The IP 265 load is determined by the COM 265 language elements in the IP 265 user program and their interconnections.

IIP 265 user program

Program which executes on the IP 265 and which controls a high-speed subprocess. The IP 265 user program is generated with the aid of COM 265, and consists of elementary functions such as logic operations, counters, timers, comparators and the like. The IP 265 user program has a defined link to the IP 265 interfaces (process I/Os and parameters).

IIP265 inputs/outputs

Inputs and outputs to process I/Os and expansion inputs for interfacing to another IP 265.

The inputs/outputs to the process I/Os are divided into 24 V inputs/24 V outputs and 5 V differential inputs (RS422).

LLanguage element COM 265 language element**O**ff-line test

COM 265 test function which allows the IP 265 user program to be tested without an IP 265 or PLC.

The user simulates input signals. Changes in the output signals or in internal intermediate states in the user program can be viewed on the programmer/PC monitor screen.

Program errors can be quickly detected and eliminated without connecting process I/Os.

On-line test

COM 265 test function with which the IP 265 user program can be tested in the IP 265. The user can control and monitor program scanning in the IP.

Using COM 265, the user can transfer control data to and read status info from the IP 265, and monitor and modify parameters.

Operand assignment list IP 265 assignment list**O**utput data

IP 265 data which is evaluated by the CPU via the process input image (PII). The data consists of the IP 265 status word and output parameters from the IP 265 user program, if any.

Output parameters Parameters

Parameters

Data interchanged between the IP 265 user program and the CPU application program. The IP 265 user program can be supplied with parameters. The IP 265 supplies IP 265 user program-dependent output parameters which can be postprocessed by the CPU application program.

An input parameter can be e.g. the preset value for a timer, an output parameter can be e.g. a count.

Program scan time

Time which passes while the FPGA processes the program. For pure binary logic operations, the program scan time is 1/128 kHz or 7.8 μ s. Each retentive language element between an input and an output in a segment increases the program scan time by 7.8 μ s.

Response time

Time between a signal state change at an input and setting of an IP 265 output. The response time is a combination of the input delay, the output delay and the program scan time.

Retentive language element

A COM 265 language element with a retention facility. Retentive IP 265 language elements include timers, counters and all set/reset functions. A retentive language element lengthens the program scan time by another 7.8 μ s.

RUN bit

Bit in the IP 265 control word. If it is set in the CPU application program, the IP 265 is set to RUN after the next CPU data cycle.

Segment

Smallest and only structural unit in an IP 265 user program. Inputs are connected to outputs in each segment.

A segment comprises COM 265 language elements, connecting lines and operands.

Standard program

IP 265 user program which cannot be read out, modified or copied. A special standard function counter can be obtained from your Siemens branch office, but you can define any self-written IP 265 user program as a standard program.

Status word

Data word via which the CPU application program can read the status of the IP 265 program. The status word provides status information as well as information on IP 265 errors/faults.

STOP bit

Bit in the IP 265 control word. If set in the CPU application program, the IP 265 is set to STOP after the next CPU data cycle.

Symbolic address

Symbolic identifier for an operand in the IP 265 user program with which the IP 265 interfaces can be referenced. It is assigned by the user, and may comprise no more than 24 characters.

Symbolic identifier Symbolic address

F Active and Passive Faults in Automation Systems

F Active and Passive Faults in Automation Systems

- Depending on the problem definition for an automation system, both **active** and **passive** faults may be **dangerous** faults. In a drive control system, for example, active faults are normally dangerous because they can result in unauthorized powering up of the drive. A passive fault, on the other hand, might not be reported at all, and may be dangerous for just that very reason.
- This distinction, and the problem-dependent division of faults into those which are and those which are not dangerous is of importance to all manner of safety considerations for a product.



Warning

Wherever faults occurring in an automation system can result in considerable damage or even endanger humans, i.e. wherever such faults could be dangerous, additional measures must be taken, or devices or methods implemented, to ensure safe operation even in the event of a fault (e.g. the use of independent limit switches, mechanical interlocks, etc.).

Maintenance procedures

If **the IP 265** should require maintenance of any kind, please observe the stipulations and directives in the accident prevention regulations VGB 4.0, particularly §8, "Permissible tolerances for active components".

Under no circumstances may the IP 265 be opened.

Repairs on automation equipment may be carried out only by the **Siemens Customer Service Department** or by **repair shops authorized by Siemens**.

G Accessories and Order Numbers

G Accessories and Order Numbers

		Order Numbers
IP 265 High-Speed Subcontrol		6ES5 265-8MA01
Product Manual (separate)	German	6ES5 998-5SH11
"	English	6ES5 998-5SH21
"	French	6ES5 998-5SH31
COM 265 Programming Package with Product Manual	German	6ES5 895-5SH11
"	English	6ES5 895-5SH21
"	French	6ES5 895-5SH31
Standard Function Counter with Instruction Manual	In three languages	6ES5 840-4SH01
Expansion cable for expansion interface	20 cm	6ES5 725-2AC01
Connecting cable for position encoder with open end	5 m	6ES5 706-8BF01
	16 m	6ES5 706-8CB61
	32 m	6ES5 706-8CD21
Connecting cable for SIEMENS position encoders (6FC9 320-...)	5 m	6ES5 706-7BF01
	16 m	6ES5 706-7CB61
	32 m	6ES5 706-7CD21
PS 931 power supply module AC 115 V/230 V; DC 24 V; 2 A		6ES5 931-8MD11
Memory submodule (EPROM)	8 Kbytes	6ES5 375-8LA11 (Prog. No. 440)
Memory submodule (EEPROM)	4 Kbytes	6ES5 375-8LC21 (Prog. No. 211)

Index

Index

A

Address	
- Direct	9-30, 9-31
- Symbolic	9-30, 9-31
Addressing	
- 5 V differential inputs	5-2
- 24 V outputs	5-2
- Control word	5-5
- Expansion inputs/outputs	5-3, 5-6 - 5-8
- Input parameters	5-3, 5-6, 5-8
- Output parameters	5-2
- Status word	5-5
Addressing operands	
- Rules for	9-29
AND operation	9-12
Assignment line	9-30, 9-40
- Input	9-40
Auxiliary error	4-10
- Bit	4-10

B

Basic functions	
- COM 265	8-5
Binary bit constants	9-10
Binary connection	9-7
Binary input	
- Deleting	9-45
- Inserting	9-42
- Inverting	9-43
Binary logic operation	9-12
Binary output	
- Inverting	9-43
Binary scaler	9-14

C

Cable shielding	3-1
Cabling conditions	3-1
Clock delay	9-16
Clock pulse generator	9-22
Clocking	
- IP 265	4-2, 4-5
- Simulator	10-17
COM 265	1-3
- Basic functions	8-5
- Defaults	9-1
- Hardware prerequisites	8-1
- Hierarchical structure	8-8
- Quantity framework restrictions	9-28
- Software prerequisites	8-1

COM 265 language elements	9-6, 9-11f.
COM 265 screen forms	
- Header line	8-4
- Input/output area	8-4
- Menu line	8-4
- Message line	8-4, 8-10
- Screen layout	8-4
Comments	9-7
Comparison functions	9-25
Compiling	9-48
Configuring data	9-4
Connecting element	9-26
Connecting line	9-6
Connector	9-7, 9-26
- 24 V load voltage	2-2, 2-5, 3-4
- Input	9-38
Constants	9-10
Control bit	4-8, 4-10, 6-1
Control word	4-7, 4-8
- Addressing	5-5
- Format	4-8
Count value	4-12, 9-23
Counter	
- Constants	9-10, 9-23
- Functions	9-23f.
- Status	4-12
CPU cycle	
- Data cycle (DCyc)	6-3
- Program cycle (PCyc)	6-3
CSF representation	4-1
- Simulator	10-3
Cursor control	8-15
- Automatic	9-36

D

Data cycle (DCyc)	
- CPU cycle	6-3
Data transfer time	6-4
Debouncing	2-3, 9-4
- Time	4-6
Delay times	4-6
Delete	
- Binary input	9-45
- IP 265 user program	11-11
- Language element	9-44
Differential inputs (5 V)	
- Addressing	5-2
- Pin assignments	2-4
- Wiring	3-3

Dimension drawing	C-1		
Displays			
- IP 265 utilization	8-10, 9-49, 9-50		
- LEDs	3-6, A-1		
- Status word	4-9f., A-2		
Distributor	9-26		
Down counter	9-24		
E			
Edge flags (Trigger)	9-15		
Error indications	4-10		
- Auxiliary error	2-2, A-1, A-2		
- Group error	4-10		
- Load error	4-10		
- Memory submodule evaluation	4-10, 4-11		
- Missing 24 V load voltage	3-6		
- Module fault	4-10		
- Short circuit	3-6		
- Wirebreak	3-6		
Error messages			
- COM 265 operator control	8-13, 8-14		
Expansion			
- Cable	3-3, 12-5		
- Interface	1-3, 2-1, 2-5		
- IP 265	1-2, 12-1f.		
- Voltage supply (24 V)	12-5		
Expansion inputs/outputs	2-5		
- Addressing	5-2, 12-6		
- Wiring	3-3		
External I/O bus	2-1		
F			
File functions	11-9		
Footer file	9-1, 9-2		
FPGA	2-1, 4-1		
G			
Group error bit	4-10		
Group error/fault	4-10		
H			
Hard STOP	4-11, A-2		
Hardware prerequisites			
- COM 265	8-1		
Header	8-4		
Help			
- Line	8-10		
- Menu	8-10, 8-11		
- Windows	8-10, 8-11		
I			
I/O bus, external		1-2, 2-1, 2-2	
Input			
- Bar		9-6	
- Data		4-7, 5-4, 6-3	
- Delay		4-4	
- Fields		9-33	
- Parameters		4-7, 4-12	
Input/output area		8-4	
Inputs			
- 5 V differential inputs		2-2, 2-4	
- 24 V inputs		2-2, 2-3	
- Expansion inputs		2-2, 2-5	
Installation		8-2	
Interface		1-2, 2-2	
- 5 V differential inputs		2-1, 2-4	
- 24 V inputs		2-1, 2-3	
- 24 V outputs		2-1, 2-3	
- Expansion inputs/outputs		1-3, 2-1, 2-5	
- External I/O bus		1-2, 2-2	
- Process		1-2, 2-1	
Inverting		9-43	
IP 265 user program		4-1f.	
- Compile		9-48	
- Copy		11-10	
- Delete		11-11	
- Display directory		11-12	
- Load		7-5, 7-7, 11-2	
- Print		11-8	
- Read from memory submodule		11-6	
- Rename		11-13	
- Storing on memory submodule		11-4	
K			
Keyboard layout			
- PC keyboard		D-1	
- Programmer keyboard		8-16	
L			
Language element		9-6, 9-11	
- Binary logic operations		9-12	
- Binary scaler		9-14	
- Clock delay		9-17	
- Clock pulse generator		9-22	
- Connecting element		9-26	
- Connectors		9-26	
- Conversion		9-43	
- Comparison functions		9-25	
- Counter functions		9-23f.	

Language element	9-6, 9-11	Parameter interchange	6-2, 6-3
- Deleting	9-44	PC keyboard	D-1
- Distributor	9-26	Pin assignments	
- Edge flag	9-15	- 9-pin sub D plug connector	2-3
- Inserting	9-42	- 9-pin sub D socket connector	2-3
- Latching	4-5	- 15-pin sub D socket connector	2-4
- RS flipflop	9-13	- Interface for 5 V differential	
- Timer functions	9-18f.	inputs	2-4
LEDs	2-6, 3-6, A-1	- Interface for 24 V inputs	2-3
Load		- Interface for 24 V outputs	2-3
- From memory submodule	7-7	PLC	
- IP 265	7-3	- Footer file	9-1, 9-2
- Via external I/O bus	7-5, 7-6, 11-2f.	- Printer file	9-1, 9-2
Load voltage (24 V)	2-5, 3-4	Presets/defaults	9-1
- Connectors	2-2, 2-5, 3-4	Print	11-8
- Error display	3-6	- File	9-1, 9-2
- Voltage supply	3-4	Process image of the inputs (PII)	4-7, 5-4, 6-1
Logic field	9-6	Process image of the outputs (PIQ)	4-7, 5-4, 6-1
M		Program cycle (PCyc)	
Margin bar	9-6	- CPU cycle	6-3
Memory submodule	1-2, 2-2, 2-6	Program execution	
- Evaluation	4-10, 4-11, A-2	- Parallel	4-3, 4-3
- Permissible	G-1	- Sequential	4-3, 4-3
Menu line	8-4	- Time	4-2, 4-4, 4-5
Message line	8-4, 8-10	Program name	9-1, 9-2
Module address	5-4, 5-5	Programmer keyboard	8-16
O		Pulse evaluation	
Off-line simulation	7-2, 7-3, 10-1f.	- 1-fold	13-14
On-line test	7-2, 7-4, 10-20f.	- 2-fold	13-14
Operand	9-6, 9-7	Q	
- Assignment list	9-30, 9-31	Quantity framework restrictions	
- Comments	9-7, 9-31	- IP 265	9-27
- Input	9-35f.	- COM 265	9-28
- Input field	9-33	R	
- Permissible	9-8, 9-9	Response time	4-4
Operating status	7-8f., A-1	Retentive language elements	4-5
OR operation	9-12	RS flipflops	
Output		- Reset dominant	9-13
- Bar	9-6	- Set dominant	9-13
- Data	4-7, 5-4, 6-3	RUN bit	4-8
- Delay	4-4, 4-6	RUN LED	2-6, A-1
- Parameters	4-7, 4-13	S	
Outputs		Screen layout	8-4
- 24 V outputs	2-2, 2-3	Segment	4-2, 4-3, 9-6
- Expansion outputs	2-2, 2-5	Segment comments	9-7
P		- Entering	9-46, 9-47
Parameter	4-7, 4-12, 5-1	Segment name	9-6
Parameter initialization		- Entering	9-46
- Timers	9-32	Service functions	11-1
		Set/reset functions	9-13f.

Shielding	3-1		
Short circuit	3-6		
Signal state display			
- 24 V inputs/24 V outputs	2-6		
Simulator	10-1f.		
Simulator settings			
- Generate	10-6		
- Reset	10-14		
- Symbols	10-15		
Slot	3-1		
- Address	5-4, 5-5		
Software prerequisites			
- COM 265	8-1		
Standard function counter	14-1		
Standard program	9-1, 9-2, 14-1		
Status display	2-2, A-1, A-2		
- COM 265 control	4-9		
- Indicator	4-10		
- Load	4-10, 7-6, 7-7		
- Memory submodule evaluation	4-10, 4-11		
- Operating status	4-9		
Status word	4-7, 4-9f.		
- Addressing	5-5		
- Evaluation	6-2		
- Status and error indications	4-10, 4-11, A-2		
- Structure	4-9f.		
STOP bit	4-8		
STOP LED	2-6, A-1		
Sub D plug connector	2-2		
Sub D socket connector	2-2		
Submodule receptacle	2-2, 2-6		
Symbols			
- Simulator settings	10-15		
Symbols editor	9-31		
- Invoking	11-2		
T			
Technical specifications	B-1		
Text format			
- Simulator	10-3, 10-18		
Time constants	9-10, 9-18		
Time response			
- Simulator	10-2		
Time value	4-12, 9-18		
Timer functions	9-18f.		
Timers			
- Parameter initialization for	9-32		
U			
Up counter	9-24		
Utilization	9-50		
- Display	8-10, 9-49		
V			
Value range			
- Count value		4-12	
- Time value		4-12	
Voltage supply			
- 24 V DC		3-4	
- Expanded IP 265		12-5	
W			
Wirebreak		3-6	
Wiring			
- 5 V differential inputs		3-3	
- 24 V inputs		3-2	
- 24 V outputs		3-2	
- Expansion inputs/outputs		3-3	
- Test		7-2, 7-3, 10-23	
X			
XOR operation		9-12	

