SIEMENS

SIMATIC S5

STEP 5 Version 6.6

Manual

C79000-G8576-C820-01

Contents

Safety Guidelines

This manual contains notices which you should observe to ensure your own personal safety, as well as to protect the product and connected equipment. These notices are highlighted in the manual by a warning triangle and are marked as follows according to the level of danger:

Note

draws your attention to particularly important information on the product, handling the product, or to a particular part of the documentation.

Qualified Personnel

Only **qualified personnel** should be allowed to install and work on this equipment. Qualified persons are defined as persons who are authorized to commission, to ground, and to tag circuits, equipment, and systems in accordance with established safety practices and standards.

Correct Usage

Note the following:

⚠ **Warning**

This device and its components may only be used for the applications described in the catalog or the technical description, and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.

This product can only function correctly and safely if it is transported, stored, set up, and installed correctly, and operated and maintained as recommended.

Trademarks

SIMATIC® and SINEC® are registered trademarks of SIEMENS AG.

Siemens AG
Automation Group
Industrial Automation Systems
Postfach 4848, D-90327 Nürnberg

Disclaimer of Liability

We have checked the contents of this manual for agreement with the hardware and software described. Since deviations cannot be precluded entirely, we cannot guarantee full agreement. However, the data in this manual are reviewed regularly and any necessary corrections included in subsequent editions. Suggestions for improvement are welcomed.

Technical data subject to change.

Siemens Aktiengesellschaft
C79000-G8576-C820

Contents

Figures

Tables

Overview 1

1.1 How to Use the Manual

This STEP 5 manual is dual purpose. On the one hand, it introduces you to the STEP 5 software and on the other, it serves as source of reference for all functions provided by the software for creating, managing, testing and documenting STEP 5 user programs.

The following questions and answers will help you to make the best use of the manual for your own personal requirements.

? **You are already an experienced STEP 5 user**

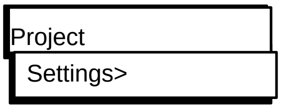
The appendix contains a brief set of operating instructions which will help you to become familiar with the menu structure without a detailed description of the individual functions.

? **Finding your way through the user's guide (Chapter 3)**

Highlighted sections of text and graphical "signposts" guide you through this chapter.

1. At the top left of a double page you will find the corresponding **menu title** from the main menu bar (e.g. Object).
2. At the top right of a double page you will see the name of the particular function described in the section (e.g. Project).
3. In the left margin you can see the menu options you select to activate the currently described function.

Example: making the settings for a project



Project

Settings>

4. (→ *Project, Settings,*). These pointers indicate that you can find further information in this section. The terms following “→” are always listed in the contents or in the index.
5. Keys such as ***ESC, Return*** or ***Insert*** are printed in italics and bold face.

? You want to run the STEP 5 software on a programmer (PG) or personal computer (PC)

You must distinguish between the following situations

a) If you have just acquired the STEP 5 software along with a new PG, it is already installed on the PG. You only require a few steps to activate the program (→ *Getting Started*)

b) You have acquired the STEP 5 software as an upgrade package or as a PC package. In this case, install the software on your PG (PC) following the instructions in the “Product Information” and start up the software.

In both cases, you will see start-up screens displayed which must be completed based on the information in the Product Information. The first STEP 5 menu is then displayed.

? You want to create a user program with the STEP 5 software for the first time

Chapter KEIN MERKER leads you step by step through the process of writing a program based on the simple example of a carwash. Using the complete example supplied as an STL program with the package, you can check that your first programming session was successful.

? You need to decide about the data management system for your user data

If you use the S5-DOS/MT operating system, you can choose between two data managers (S5-DOS and BTRIEVE). Read Chapter KEIN MERKER for an explanation of the advantages of one against the other. Make sure you decide before starting to write your user program.

? You want to get an idea of the operating elements you use to move through the STEP 5 software

Section 3.1 provides you with this information. It is advisable to read this section before beginning to work with STEP 5.

? You are writing, testing, managing or documenting a STEP 5 user program and are unsure about using a function

The first source of help is the STEP 5 software itself. If you cannot find an answer, turn to the menu description in the sections 3.2 to 3.8.

? You have user programs you created under S5-DOS (PCP/M) and you want to continue working with them under S5-DOS/ST with the new STEP 5 version

Convert these programs using the copy function “PCP/M file - Copy” in the “Object” menu (→ *Section 3.2.4*). You can then use all the processing functions.

1.2 Product Overview

Operating System

STEP 5 software from release 3.0 can be used on a PG 7xx with the following operating systems:

- S5-DOS/ST (ST = Single Tasking using MS-DOS),

and on an AT compatible PC with a special operating system package (STEP 5 programming package for PCs).

Information about installation and the functions of the operating systems can be found in the relevant manuals.

1.2.1 New Features Compared with Previous Versions

Release V6.0 and higher differs considerably from previous STEP 5 versions. It also includes the extended functions and modifications explained briefly below.

Data/File Management

- STEP 5 can no longer be run with S5-DOS (PCP/M). You can convert existing programs so that they are compatible with the operating system you decide to use. (→ *Project, PCP/M File*).
- It is now possible to “navigate” through the DOS file system without having to change to the MS-DOS user interface (→ *Object, DOS file).*
- The files *DR.INI and *AP.INI are now managed centrally in the STEP 5 system directory (→ *Project*).

User Interface

- The user interface allows elementary functions to be selected from menus (→ *Graphical user interface*).
- You define functions in “job boxes” (→ *Graphical user interface*) which have largely replaced the command lines.
- A mouse is now available to activate and select many functions (→ *Graphical user interface*).
- The Return key corresponds to the Enter key (<OK> button in the dialog window).

- Function keys allow you to select the most important functions quickly (→ *Graphical user interface*).
- Hotkeys, with which you can select a function directly are also available. These hotkeys are indicated by a red letter in the menu name (on a black background on monochrome monitors) (→ *Graphical user interface*).
- In many situations, you can display help texts by pressing ***SHIFT F8***.

Preparations for Programming

- All the settings and selections for files and program parameters for a project have been put together in two basic “settings boxes” (→ *Project*, *Settings*).
- All the presets are stored in a file which can be reloaded making repeated setting of program parameters unnecessary (→ *Project*).
- The presets you select during one programming session are still valid when you want to program again, even after exiting STEP 5.

Documentation

- The previously independent package KOMDOK is now an integral part of STEP 5 (→ *Enhanced output*).
- Within each function the destination of the output can be selected (→ *Graphical User Interface*, *Job box*).

Editing

- All the modifications involving editing are listed below, sorted according to functions.

Connector

- Apart from the existing connector, a negated connector with the same applications has been implemented. (→ *Editor, Editing Ladder Diagrams/Control System Flowcharts).*
- You can now specify the connector and the negated connector immediately before CSF outputs.

Complex Function Elements

- The new language elements can be represented as complex function elements (→ *Editor, Editing Ladder Diagrams/Control System Flowcharts)*

- You can combine all complex function elements in a CSF segment.
- Complex arithmetical function elements have extendable inputs (function element with two inputs).
- The screen representation of the status display of complex function elements has been extended (→ *Test, Block Status).*

Cursor Display

- To indicate the current editor position, a long cursor has been introduced, its length corresponding to the length of the input field (→ *Editor).*

Block Calls

- In an empty segment, a call can be input directly using the function keys. You can append calls in existing segments with/without the automatic expand function.

New Output in CSF

- Implicit appending or inserting of a new output is possible (→ *Editor, Editing Control System Flowcharts).*

Cursor Positioning

- There are new key functions for positioning the cursor in the editing window (→ *Appendix A4, Key Assignment).*
- These cursor positioning functions can also be activated using function keys (“Extras”) (→ *Editor, Editing Ladder Diagrams/Control System Flowcharts).*

Expanding

- Whenever the expand function would otherwise be necessary, this is now performed automatically (→ *Editor, Editing Ladder Diagrams/Control System Flowcharts).*

Editing Mode

- The “correction” mode no longer exists, there are only the modes “edit” and “output”.

FB

- There are new opportunities for editing FBs:
  - They can now be programmed in LAD and CSF. With the exception of the first segment, all new language elements can now be used graphically within a segment (→ *Editor, LAD/CSF).*
  - The formal operands defined in the first segment cannot be used in a LAD or CSF segment.
  - The FB name is displayed in the “Directory” function (→ *Object, Blocks).*

DB

- You can only select the LIB no. using function keys.

Segment in a Block

- Segment functions can be selected using softkeys (→ *Editor, Editing Assignment Lists*)
  - – Copying (within a block and from a buffer).
  - – Delete, insert, append, page
  - – Empty segment possible in LAD and CSF

Function Keys

- Double function keys are now available. These reduce the number of hardkeys.

Editing the Assignment List

- The STRING assignment of the function keys is now visible in the symbols editor and is available with ***F1*** to ***F4*** (previously ***F1*** to ***F8***) (→ *Editor, Editing the Assignment List).*

Bus Paths

- Bus paths can be renamed.
- The bus paths are reduced to possible (and guaranteed) paths. If the path deviates from the preferred path, a message is displayed (→ *Management, Bus Paths*).

EPROM

- EPROM submodule programming numbers can be selected from a selection box (→ *EPROM*).

Online

- You are informed of the PLC type and the CPU number in all online functions.

Practical Application of STEP 5 - Programming Example -

2

2.1 Introduction to the Example (Control Task)

To help you get to know STEP 5 and get used to working with this software package, this section explains an application based on an example. The control task “controlling a car wash” shows you step by step how to edit, test, document and archive the required user program.

This introduction to the use of STEP 5 based on an example has the following two aims:

- to make the most important system and editing functions on the programming device available to practised users as quickly as possible and
- to provide information about planning and implementing a project using the STEP 5 tools for first time users.

The development of the required STEP 5 program to control the process is not part of this example. Nevertheless, the steps necessary to produce such a program are explained in Appendix A1, in case you would like to write the program yourself. The complete program consists of the following parts:

- an assignment list (absolute operands, symbolic operands),
- a function block with 15 segments in the “Statement List” (STL) method of representation,
- a data block,
- the organization blocks for start-up and cyclic operation of the car wash.

It is advisable to try out the steps explained in Section 2.2 on your PG. You will probably only need to edit a few segments. You can find the complete function block in the directory \S5_DATEN\DEFAULT along with all the other parts of the example program.

Brief Description of the Control Task

The following illustration shows a carwash of a type commonly found at gas stations and this is what we want to automate with the STEP 5 program.

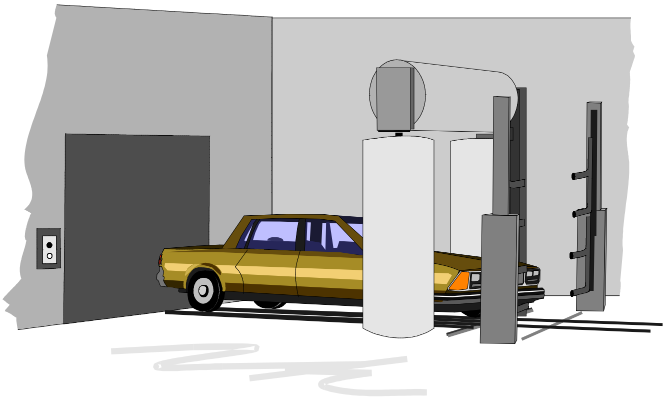


Figure 2-1 Carwash

The structure of the carwash and the steps necessary to clean the car result in the following sequence of events:

- the carwash moves to a starting position
- the car is driven into the washing position
- the door of the carwash is closed and the washing is started
- shampoo is applied, the car is washed and rinsed, wax is applied and the car is dried
- finally, the door is opened automatically and the car can be driven out.

Certain variables such as the time allowed for drying or for the wax to distribute evenly, can be modified by the operating personnel. The controller records the number of washing cycles (i.e. number of cars washed).

Conditions Necessary to Implement the Example

Based on the detailed schedule for the washing process outlined above, we can determine the “process interfaces”, i.e. the inputs/outputs for the required control system (*Figure* 2-2). By labelling the I/O signals (Signal list, *Table* KEIN MERKER), based on the verbal description of the process, the control program to implement this process can be developed (*Appendix* KEIN MERKER1).

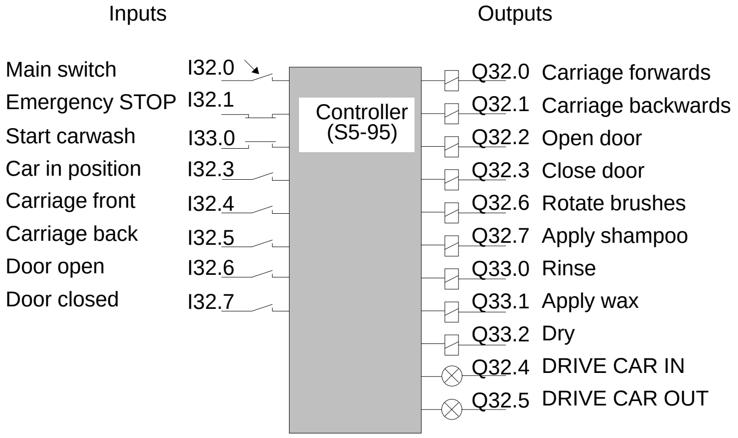


Figure 2-2 Controller with Process Interfaces

The following figure shows the hardware and software components required to implement the example. You only require the S5-95 and the simulator to test the control program.

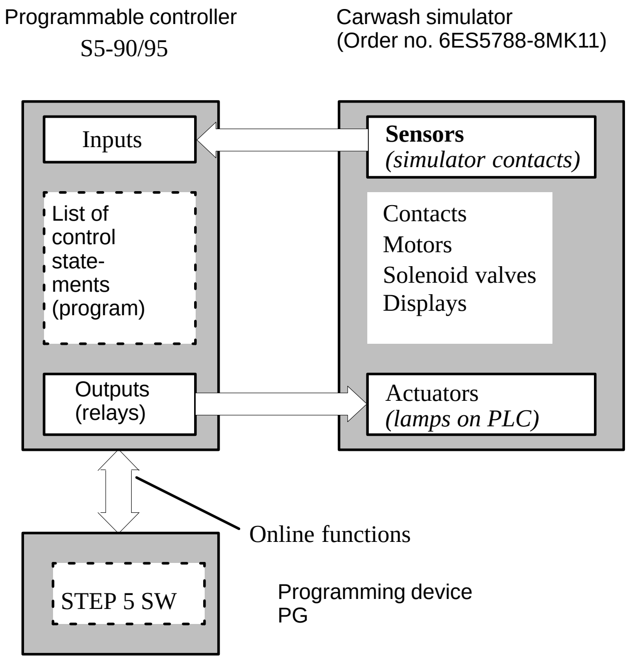


Figure 2-3 Configuration of the “Carwash” Example

2.2 Creating the Carwash Program with STEP 5

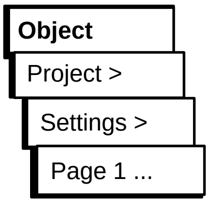
We will call the carwash control system our “project” in keeping with the STEP 5 terminology. Creating the user program on the PG can be divided into the following phases:

- setting up and opening the project
- creating the contents of the project (editing and structuring the program)
- managing and handling the project.

2.2.1 Setting up the Project

Since the operating system and the programmer start-up depend on the particular PG being used, we must start the description of the example assuming that the STEP 5 initial menu is already displayed.

Beginning with the menu selection “Object” you make all the settings and parameter assignments necessary to prepare for programming in the submenu “Project”.



1. For a new project, you select “Project, Settings, Page 1”. To select the existing project at a later date, you use ”Load project”.

   Page 1 of the input window appears with input fields for various file names. These files either have defaults or “NONAME” entered.

2. Specify the program you wish to create for the carwash by overwriting the defaults with the following names:

Working directory:	C:\S5_DATEN\EXAMPLE
Program file:	C: CARWAS**ST.S5D**
Symbols file:	C: CARWAS**Z0.INI**

The characters shown in bold face are fixed and cannot be modified.

3. You can make entries in the input fields by positioning the cursor on the file name and then pressing ***F3*** = *Select.*
4. To select the working directory
   – first press ***F3*** twice, in the file selection box. Select the subdirectory “*EXAMPLE*” under “*Dr C:*”.
   – After confirming twice with ***OK***[1] the working directory *C:\S5_DATEN\EXAMPLE* is selected.
5. Press ***F4*** to change to Page 2.

Note

If you have problems selecting directories or files in the definition and selection boxes, refer to the introductory section 3.1 in the “User’s Guide - Part 3”.

Select Mode

As long as there is no PLC connected, only “offline” is possible as the mode and is therefore preset by STEP 5.

Select Operand Representation

6. Set “Symbols” to “yes” with ***F3*** and the same for ”comments”.

The “display” parameter is set to “Sym”.

Name the Printer File

7. Overwrite the name of the printer file . . . ***DR.INI*** with our program name *“CARWAS”*.

The name is automatically entered as the name of the documentation file ...LS.INI.

Select Method of Representation

Since we want to program in “Statement List”

8. Set this parameter to STL by pressing ***F3*** repeatedly until STL is displayed.

Select Symbol / Comment Length

To simplify matters, we will leave the maximum symbol length at 8 characters. Since, however, a more detailed explanation will be helpful,

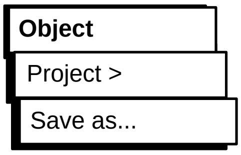
9. Change the comment length to 40 characters. You must complete this entry with the ***Return*** key.

Save the Settings

10. *You return to the menu by pressing **F8**.*

11. *After selecting “Project, Save as...” ;*

the file selection box appears in which you enter “CARWAS” under file name.



After clicking on **OK** and acknowledging the message “Destination file already on FD, overwrite?”, STEP 5 sets up the project file **CARWASPJ.INI,** which contains the program files and settings.

2.2.2 Creating the Program

Once you have specified the project by naming files and selecting parameters, we can now start entering the statements or operations in the function block and the timer and counter values in the data block.

Our intention is to show you how to make the inputs and not to work through the example to the end. We will only make the inputs until they start to become repetitive. You can copy the complete program with all the blocks and segments to your working directory from the directory C:\S5_DATEN\DEFAULT under the project name PROBSPPG.INI.

To make the program easier to read, we will work with symbolic operands in the control statements. This means that we require an assignment list before beginning editing STL.

The creation of the carwash program therefore involves the following editing steps:

- compilation of a list with the assignments of absolute operands to symbolic process signal names
- creation of the data block for process setpoints and to record the number of cars washed (i.e. number of process cycles)
- creation of a statement list in a function block to control the process.

These three steps will give you the opportunity to get to know the three most important STEP 5 editors.

Editing the Assignment List

Symbolic operands are names (e.g. “OPEN DOOR”) of the absolute operands processed by the controller (e.g. I 32.6, Q 32.2, F 10.0). So that the programmer “understands” the symbolic operands you are using, an assignment list (ASSLI) is necessary, in this case, this is edited in the symbols file with the name C:CARWASZ0.SEQ.

As the basis for creating this list, use the list of process signals (*Table* KEIN MERKER), in which you can see the assignments. Before these symbolic operands are entered in the ASSLI, they must be reduced to the maximum 8 characters selected in the settings. The use of upper case characters for the symbols makes the program clearer.

Editor
Assignment list F7

1. *Call the “assignment list” STEP 5 editor in the editor menu (or press F7).*

Below the top line containing CARWASZO.SEQ, an empty screen form is displayed with the columns “Operand”, “Symbol” and “Comment”. You have already stipulated the lengths of the fields for the symbolic operands and comments.

2. Type in the first line of the assignment list as follows:

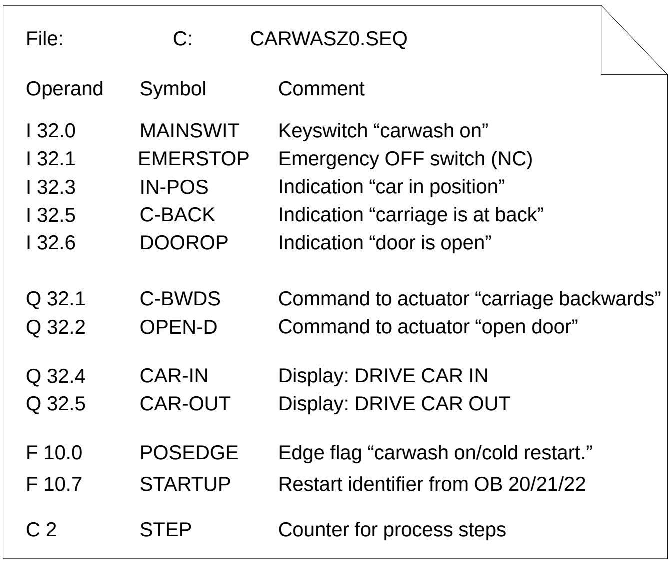
Operand	Symbol	Comment
I 32.0	MAINSWIT	Keyswitch “carwash on”

3. To do this type in the characters: I 32.0 (in the insert mode) and press ***SHIFT cursor right*** or ***TAB***.
4. Type in MAINSWIT (this field is then full, the cursor automatically jumps to the next field).
5. Type in “Keyswitch “carwash on”” and press the ***Return*** key or ***TAB***.

Figure 2-4 shows you an extract of the assignment list. Enter this list as it stands in your symbols file. To complete the editor editing session

6. Press the ***Insert*** key or ***F7*** = *OK* .

This stores the file and starts the translation. The PG generates the symbols files required by STEP 5 of the type . . . Z*.INI.



File: C: CARWASZ0.SEQ

Operand	Symbol	Comment
I 32.0	MAINSWIT	Keyswitch “carwash on”
I 32.1	EMERSTOP	Emergency OFF switch (NC)
I 32.3	IN-POS	Indication “car in position”
I 32.5	C-BACK	Indication “carriage is at back”
I 32.6	DOOROP	Indication “door is open”
Q 32.1	C-BWDS	Command to actuator “carriage backwards”
Q 32.2	OPEN-D	Command to actuator “open door”
Q 32.4	CAR-IN	Display: DRIVE CAR IN
Q 32.5	CAR-OUT	Display: DRIVE CAR OUT
F 10.0	POSEDGE	Edge flag “carwash on/cold restart.”
F 10.7	STARTUP	Restart identifier from OB 20/21/22
C 2	STEP	Counter for process steps

Figure 2-4 Assignment List (Section to be Edited)

After the translation, STEP 5 displays one of the following messages:

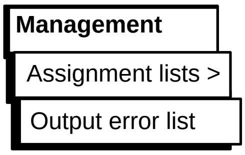
”n lines processed, no errors found” or

”error in line n” and e.g. “key not found” or

”n lines processed, x errors found”.

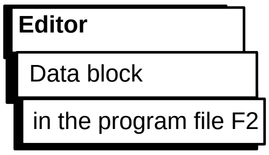
If no errors are found, you have successfully completed editing the assignment list. If an error is found, the incorrect line is displayed at the top.

If errors are indicated, display or print out the error list as follows:



1. Press ***OK*** and ***continue***.
   This brings you into the menu.
2. Under “Management”, select the submenu “Assignment lists” and “Output error list”.
3. Read the error list directly from the screen or print it out.
4. Make the corrections for the assignment list in the editor and trigger the translation again.

Editing the Data Block



1. You call the editor for creating data blocks in the menu under ”Editor” and “Data block in the program file ....” (or function key F2).

Use *Figure* KEIN MERKER in the Appendix for the contents of the data block.

Naming the DB

2. Enter the type and number of the data block to be created in the job box, in this case: DB5. Confirm this with ***OK***.

Checking the Header Line

In the header line of the empty input field, the name of the block DB5 and the program file C:CARWASST.S5D appear. The editor specifies the addresses of the data words beginning with 0.

Enter Format

3. First enter the format for the data word (KH).

If the format is “valid” the cursor jumps to the next field. If you make an illegal entry, this is rejected with the message “Input illegal”.

Inputting a Data Value

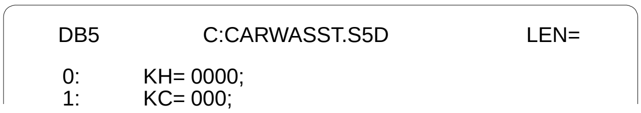
4. You must now type in the numerical value in the preset format, keeping to the corresponding range of values.

Illogical values are not accepted. The cursor will not move even if you press the ***Insert*** key.

Inputting Further DWs

The next DW field (following line) is displayed with the same format. If you require a different format:

5. Go back with ***cursor left*** and enter the required format.



```
DB5              C:CARWASST.S5D              LEN=

 0:        KH= 0000;
 1:        KC= 000;
```

6. Type in the remaining data words as shown in *Figure* KEIN MERKER.

Correcting in the Data Field

Delete Character

– Position the cursor on the character and press ***DEL***.

Insert Character

– Position the cursor on the character you want to insert a character before and press ***expand horizontal***, if necessary several times.

Delete Line

– Position the cursor in the format field of the line you want to delete and press ***DEL***.

Insert Line

– Position the cursor in the format field of the line you want to insert a line before and press ***expand vertical***.

Typing in DW Comments

You can type in or overwrite the comments for the data words in upper or lower case letters with up to a maximum of 32 characters.

7. Position the cursor in the comment field with ***SHIFT cursor right***. Move to the next line with ***cursor down***. Insert /delete characters as in the data field (see above). Insert/delete comment lines using the function keys ***F1*** *= Expand DC* and ***F2*** *= Delete DC.*

Entering the block title

To enter the title “Carwash: counters/timers”

8. *Type in the text after pressing* ***SHIFT F6*** *or* ***COM****.*
9. *Press the* **Insert** *key to return to the DW editing area.*

Writing the Block Comment

You call the editor for the block comment by pressing ***SHIFT F7*** = *Comment* or ***COM*** twice.

10. Type in the text from *Figure* KEIN MERKER, completing each line with the ***Insert*** *key.*

Making Corrections in the Block Comment

To try out the "insert/delete" functions in this editor. Position the cursor on the "c" of controller in the second line and press ***F1*** = *Insert.*

The editor is in the insert mode. The softkey label changes to ***F1*** = *Overwrite*, i.e the selectable mode is displayed and insert is set.

Type in "Simatic". The text is inserted at this point. You return to the overwrite mode with ***F1*** = *Overwrite.*

Now position the cursor on the "S" of Simatic and press ***F2*** = *Delete,* move the cursor to the "c" of controller and ***F2*** = *Delete* again.

The word you inserted is now deleted.

Completing the Comment

Complete the comment with ***F8*** = ***Return*** and ***Insert*** or ***Insert*** twice.

Inputting the LIB. No.

As the final step in the editing session, specify a library number to identify the block (e.g. DB version).

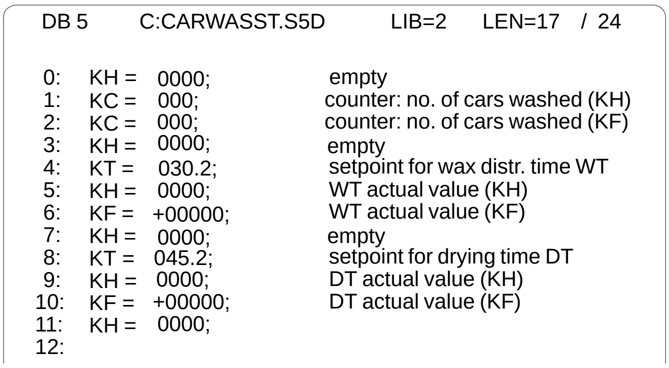
11. Press ***SHIFT F2*** = *Lib no.,* the cursor jumps to the LIB field, type in the LIB number, in this case "2". Exit the field with the ***Insert*** or the ***Return*** key.

Terminating the Editing Session

Once your screen contains the information described above:

12. Complete editing the DB by pressing the ***Insert*** key. If the message "DBn Already in file, overwrite?" appears, confirm with ***yes***.

Your inputs or modifications are now edited and saved (in some cases the messages must be confirmed twice).



```
DB 5      C:CARWASST.S5D       LIB=2    LEN=17   / 24

 0:   KH =  0000;          empty
 1:   KC =  000;           counter: no. of cars washed (KH)
 2:   KC =  000;           counter: no. of cars washed (KF)
 3:   KH =  0000;          empty
 4:   KT =  030.2;         setpoint for wax distr. time WT
 5:   KH =  0000;          WT actual value (KH)
 6:   KF = +00000;         WT actual value (KF)
 7:   KH =  0000;          empty
 8:   KT =  045.2;         setpoint for drying time DT
 9:   KH =  0000;          DT actual value (KH)
10:   KF = +00000;         DT actual value (KF)
11:   KH =  0000;
12:
```

Editing a Function Block

Editor

STEP 5 block >

in the program file F1

1. You call the editor for creating STEP 5 blocks in the editor menu under “STEP 5 block, in the program file”.

The job box is then displayed again.

2. Here you can enter the type and number of the block you want to create in the job box.

Naming a Block

The possible block types are available in the selection box, and you can display this as follows:

3. Press ***F3*** *= Select.*
4. Enter the type and an unused number for the block to be created in the block field of the selection box, in this case FB 5, and complete your entry with ***OK***.

   STEP 5 enters the information in the job box.
5. Mark the options
   – ”Confirm before overwriting “ and ”
   – ”Update seq. source file”

   with ***yes*** *and then click on* ***OK*** *again.*

The input field of the editor is then opened.

Entering a Block Name

The header line contains the block name (FB 5), the program file (C:CARWASST.S5D) and the length of the block with its header (LEN=0). The cursor is positioned in the “Name” field, where 8 characters are available to name the function block.

6. Type in *CARWASH* and press the ***Return*** key.

The cursor jumps to the field “Decl: ...” which is only significant for function blocks in which parameters can be assigned.

7. Exit this field by pressing the ***Return*** key again.

Entering Statements for Segment 1

The cursor is now positioned in the input field for the first statement. Take out the printed program excerpt from Appendix A1 (step 5).

8. Type in the statement in segment 1: C DB 5 and then press ***SHIFT cursor right*** or ***TAB cursor right***.

The cursor is positioned in the field for the statement comment.

9. Type in the text “call DB 5 (timer/counter values)” and then move on to the next statement field by pressing the ***Return*** key.

Typing in the Segment Title

Segment 1 does not contain any further statements, however, the segment title has not yet been entered.

10. Press ***COM*** and ***SHIFT F6*** = *Title* and type in “Prepare program execution”. You exit this field again by pressing the ***Return*** key or ***Insert***.

Typing in Statements for Segment 2

We now move on to segment 2.

11. Press ***Segment end*** *(***)*

The cursor is positioned in the first statement field of segment 2.

12. Type in the statements and statement comments based on Step 5 in Appendix KEIN MERKER. Write the operands using the symbolic names specified in the assignment list. These must be preceded by a hyphen in the statement field.

You can type in all the entries in the statement section without blanks. However, symbols defined in upper case letters must be written as upper case letters.

Correcting the Symbols File

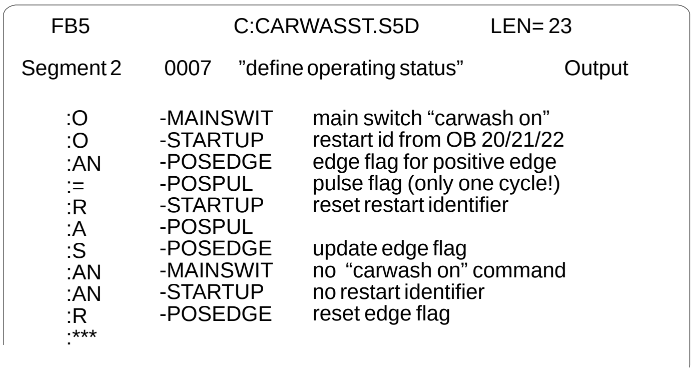
In the 4th and 6th statement lines you will notice that when you type in -POSPUL, the cursor jumps back to the hyphen and cannot be moved out of the field. This symbol has not been assigned to an operand (message: No assignment, symbol not defined), and this must be corrected.

13. Instead of -POSPUL, type in the formal operand F10.1 to be able to continue editing the segment which is finally completed with the ***Insert*** key.
    Reply to the message: Enter changed segment? with “yes”. You then change to the output mode.
14. In the “output” mode of the editor, position the cursor on the 4th statement again and press ***F1*** *= Disp symb* to call the symbols editor.

From the symbols file ...*Z0.INI, the sequence of statements with symbolic assignments is displayed with the cursor marking the formal operand F 10.1. Complete this line with the symbol ”POSPUL” and the corresponding operand comment “pulse flag for F 10.0 (only 1 cycle!)”.

15. Press ***F2*** *= Edit symb* and after typing in the symbol and comment, press ***F2*** *= Insert.* Complete the correction by pressing ***F8*** *= Return.*

When you return to the block editor, segment 2 should appear as shown below.



```
   FB5                C:CARWASST.S5D         LEN= 23

Segment 2    0007   ”define operating status”            Output

    :O        -MAINSWIT     main switch “carwash on”
    :O        -STARTUP      restart id from OB 20/21/22
    :AN       -POSEDGE      edge flag for positive edge
    :=        -POSPUL       pulse flag (only one cycle!)
    :R        -STARTUP      reset restart identifier
    :A        -POSPUL
    :S        -POSEDGE      update edge flag
    :AN       -MAINSWIT     no  “carwash on” command
    :AN       -STARTUP      no restart identifier
    :R        -POSEDGE      reset edge flag
    :***
```

Correcting Statements

You make corrections in the statement and comment field in the same way as when editing the data block. There is, however, one difference: the delete and insert line functions affect the whole line. To delete a line, position the cursor on the appropriate statement colon.

Writing the Segment Comment

Call the segment comment editor as follows:

16. Press ***SHIFT F6*** = *Seg com* and ***SHIFT F7*** = *Comment* or press ***COM*** twice.

Under the $ character with the segment number, you can now write your comment text. (Based on the printout of the program at the end of Appendix KEIN MERKER1).

17. Type in the texts for segment 1 and segment 2, completing each line with the ***Return*** key. You return to the block editor with ***F8*** = *Return.*

Statements for Segment 4 and Segment 5

Once you have pressed **Segment end**, the cursor is positioned in the first statement line of segment 3. You can now type in the statements and comments for segment 4 and segment 5. We have skipped segment 3 and will insert it later.

One special feature in segment 4 is the program branch with a conditional jump to the second statement. The jump label CONT must be positioned at the destination of the jump to mark the re-entry before the statement colon.

18. *Press the **cursor left** key twice and type in the jump label.*

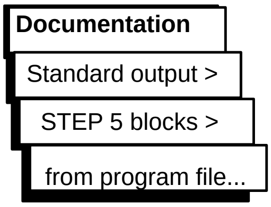
Inserting Segment 3

19. Use ↓↓↓ = scroll forward or ↑↑↑ = scroll back to page to segment 3 and press ***F5*** = *Seg fct*
and then press ***F5*** =*Insert* again.

After pressing ***F1*** = *New*, the cursor is positioned in the first statement of the newly inserted and still empty segment.

20. Edit the segment and complete it by pressing the ***Insert*** key and confirming the system prompts.

2.2.3 Documenting the Program



You can now print out the program section in FB 5, the data block and the assignment list. The printer file has the default name NONAMEDR.INI in page 2 of the settings. Overwrite this with CARWASDR.INI.

Change to the “Documentation” menu and select the standard output of STEP 5 blocks.

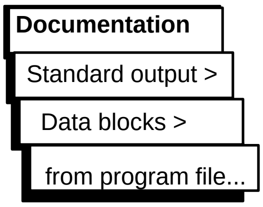
As you will see in the job box, STEP 5 provides you with the possibility of specifying blocks and segments.

1. Enter “FB 5” from your program file in the job box.
2. Under the options, select the address representation “word-oriented” and the printout type “standard”.
3. The printout is triggered with ***OK***.

The printout contains the following elements for each segment:

- the segment title and segment number
- the statements section with line comments
- the names of the operands in the assignment list.

Your printout of the program CARWASST.S5D should now correspond to the program excerpt shown in Appendix KEIN MERKER (Section 5) apart from the symbols names.



Follow the same procedure to obtain a printout of data block ”DB 5” and the assignment list “CARWASZ0.SEQ” by selecting the appropriate submenu items.

You can print out other existing blocks by pressing ***F3*** = *Select* and selecting a block in the selection box.

If you have not connected a printer to your PG, you can output the documentation to a file and print out the blocks from a different PG/PC.

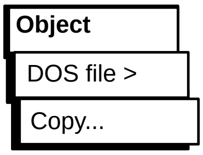
In this case, mark “Documentation to file” in the job box for the settings (Page 2) and specify the file name “CARWASLS.INI”.

This file name is automatically entered in the settings on page 2 “Documentation to file”.

2.3 Transferring Files, Blocks and Segments

We interrupted the editing of the carwash program at the 5th segment and will now add the missing sections from the supplied program. This will familiarize you with the directory, transfer, copy and delete functions in STEP 5.

The complete program is located under the name PROEXA... in the directory \S5_DATEN\DEFAULT. To transfer the file, change over to the DOS file functions as follows:



1. Select *"DOS file"* and *"Copy"* in the Object menu.

The job box "Copy DOS files" is displayed. Here, you select the source and destination directory for the transfer and assuming that you do not want to transfer all the source files displayed in the middle window, copy the files belonging to the program one after the other in the "single" mode.

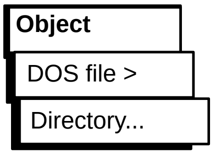
2. First check that the directories are correctly selected.
   Source drive: C:\S5_DATEN\DEFAULT
   Destination drive: C:\S5_DATEN\EXAMPLE

We want to copy the files PROEXA*.* . To do this:

3. Mark "all" in the "Copy mode" window and select "yes" in the "Confirm before overwriting" window.
4. Trigger the transfer by clicking on ***Transfer*** or pressing the ***Return*** key.

If you have selected "confirm before overwriting", STEP 5 displays the prompt "File already exists, overwrite?" if you repeat a copy procedure.

5. Confirm the prompt with **yes** and exit the box after the transfer with ***ESC*** *= Exit.*



In the "DOS files - directory" menu check that all the PROEXA.. files have been copied as follows:

set the directory C:\S5_DATEN\EXAMPLE\ under "Drive/dir.

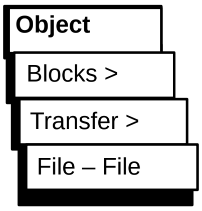
Apart from the files of the CARWAS... program, the PROEXA... files must also be entered.

Now that both programs are in the working directory, you can add the missing program sections to the incomplete program by

1. transferring the missing segments,
2. replacing the incomplete block FB5, by FB10 containing the complete carwash program and renaming it as FB5,
3. transferring the missing organization blocks (the data blocks are identical).

1. Transferring Segments

Segments can only be transferred between blocks in the same program. This means that the function block FB10 must be transferred from the program PROEXAST.S5D to our program CARWAS... .



To transfer blocks, select “Transfer blocks” and “File - file”, STEP 5 then displays a file selection box in which you specify the following:

1. as source C:PROEXAST.S5D
   –>*\S5_DATEN\EXAMPLE\* and

   as destination C:CARWASST.S5D
   –>\S5_DATEN\EXAMPLE.

When you press ***F3***, STEP 5 displays the files located in the working directory.

2. Under “Selection” and “block list” enter the block you wish to transfer (here, FB10) .

After clicking on ***Transfer*** or pressing the ***Return*** key, STEP 5 displays the prompt “Write preheader to FD?”.

3. Reply with ***yes***.

This is followed by the prompt “Transfer comments as well?”

4. Confirm the message with “yes”.

Note

The messages “FC10 Already in file, overwrite?” and “FBDO.010 Already in file, overwrite?” do not appear the first time you transfer.

---

5. Confirm the message “block(s) transferred!” with ***OK*** and exit the job box with ***ESC*** = *Exit.*

Check the transfer in the block directory in the program file.

Object
Blocks >
Directory >
in the program file...

1. Select “Blocks, Directory, in the program file” in the Object menu or use ***F3*** in the selection box (Directory file:settings).

Click “all blocks” (if not already selected as default) so that

2. after clicking ***Output*** (or pressing the ***Return*** key or ***Insert*** key)

a list of the blocks in the program file CARWAS... is displayed on the screen. By marking the corresponding selection, you can also output this list on the printer or to a file.

To transfer segments

Editor
STEP 5 blocks >
in the program file...

1. G*o* into the block editor and select FB10 in the job box.
2. Move the cursor to segment 6 using ↓↓↓ = scroll down or the + key.
3. Press ***F5*** = *Seg fct* and ***F4*** = *File.*
4. With ***F8*** = *Return* and ***ESC*** = *Exit* you can now exit FB10.

A copy of segment 6 is loaded in the system buffer. To transfer this to FB5

5. Select FB5 in the block editor and move the cursor to segment 5 at the end of the program.
6. Press ***F5*** = *Seg Fct* and ***F6*** = *Append.* Then press ***F2*** = *Buffer* to append segment 6 to the program CARWAS.... .
7. Complete the operation with ***F8*** = *Return* and ***F7*** = *OK.* Reply to the STEP 5 prompts with ***yes***.

You then exit the editor. Repeat the transfer procedure for segment 7.
As you will see, not all the operands in the new segments have been written as symbols. This is due to the incomplete assignment list in the previously edited program section. To correct the situation, proceed as follows:

*Go to Page 1 of the project settings and enter PROEXAZ***0.INI** *as the symbols file. Then save with* **F6.**

Since the block editor can now access the complete assignment list of the supplied program, the operands in segments 6 and 7 are also displayed in symbolic form.

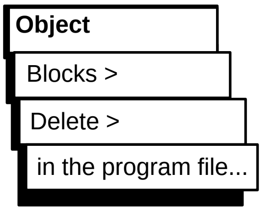
You can check this by calling FB5 again in the block editor.

With this procedure, you can append or insert segments from other blocks into the program file. To transfer and extend larger program sections, this method is, however, time-consuming.

2. Transferring and Renaming Blocks

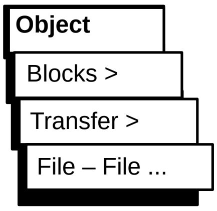
To replace FB5 in the program CARWAS... with FB10 completely,

- FB5 must first be deleted including the comments
- and then FB10 renamed as FB 5.



1. To delete FB5, select “Blocks, Delete” in the Object menu, enter “FB5” in the job box .
2. After you press ***Delete***, the system prompts “*Delete comments as well?*”. Confirm this prompt with ***Yes*** *and the message “block(s) deleted!” with* ***OK****.*

If you check the block directory, you can make sure that FB5, FC5 and FBDO.005 have been deleted.



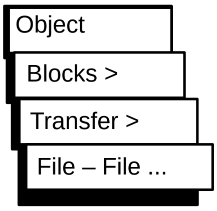
1. To rename FB10, select “Blocks - Transfer” in the Object menu and then enter or select
   – destination file C:CARWASST.S5D and
   – mark “rename block” (X), [FB10] to [FB5]
2. Click on ***Transfer*** and confirm the system prompts with ***yes***.

When you check the block directory, you will see that there is a new FB5/FC5 along with FB10/FC10.

In the editor, check that the new FB5 is complete with 15 segments, symbolic operands and all comments.

3. Transferring the Organization Blocks

To complete our program containing FB5 and DB5 the missing organization blocks must also be transferred.



1. To transfer the OBs, select “Blocks, Transfer” in the Object menu and enter the source file PROEXA... and the destination file CARWAS... in the job box.
2. Mark all “OBs”. When you click on ***Transfer***, the system displays the message “Transfer comments as well” which you confirm with OK and then “Blocks transferred”, which you again confirm with ***yes***.

The unconditional jump operation in OB 1 must now be changed to JU FB5 and the data block call C DB 10 must be changed to C DB 5 in FB 5, following which the CARWAS... program contains all the blocks required for the controller.

2.4 Checking and Modifying the Program

Apart from the editing functions, STEP 5 provides a series of functions with which you can check and document the user program and rename operands. You can now try out some of these functions on the carwash program.

Cross References

STEP 5 stores cross references to statements containing the same operand (even in other blocks) in the XRF file (*XR.INI). You can generate this file

Management
Generate XRF

by selecting”Generate XRF” in the management menu.

The XRF file is then entered in Page 1 of the “settings”. You can now display the cross references for each operand in the block editor.

1. Call FB 5 in the block editor and position the cursor in segment 2 on statement “:O -STARTUP”.
2. Press ***F2*** *= Reference* and once again ***F2*** *= Disp XRF.*

The cursor now flashes under F 10.7, the operand with which the cross references will be displayed.

3. Confirm with the ***Insert*** key.

A table of cross references for the selected operand is now displayed (*Figure* 2-5). This table contains all the points in the program at which the relevant operand is “addressed”. The cursor is positioned on the first block reference “OB20 :1/AN”.

4. Press ***F2*** *= Jump.*

The organization block OB 20 is displayed. If necessary, you can change to the editing mode and make modifications. To return to the table:

5. Press ***F2*** twice and the ***Return*** key.

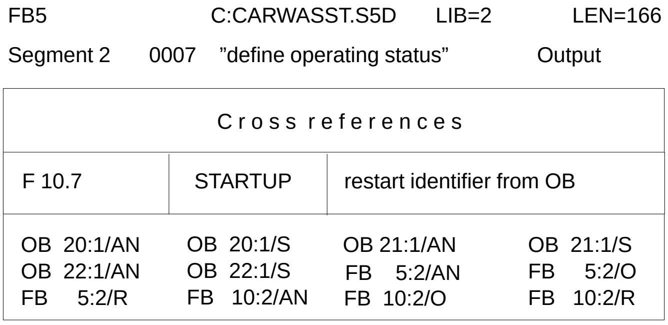
To return to FB5 press F2 to change back to OB 20 then

6. Press ***F2*** *= Reference* followed by ***F5*** *= Orig blk.*

You can repeat the jump to a referenced block

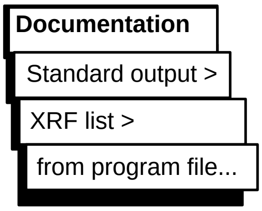
by positioning the cursor on FB10:2/AN in *Figure* 2-5 and pressing ***F2*** = *Jump*.

SEG 2 in FB10 is then displayed.

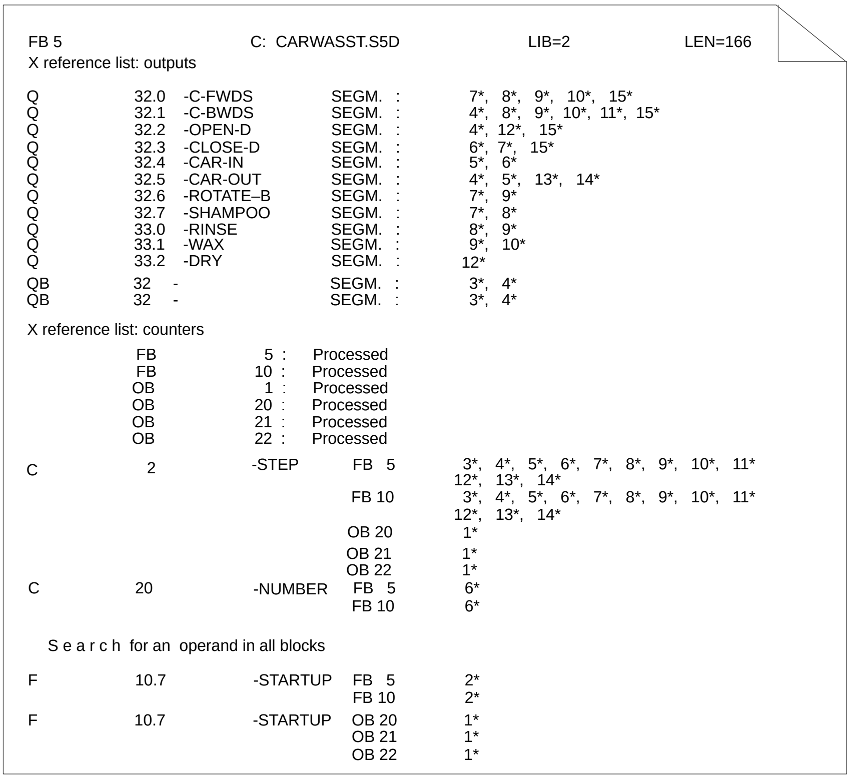


```
FB5                C:CARWASST.S5D     LIB=2          LEN=166
Segment 2    0007   "define operating status"         Output

                  C r o s s   r e f e r e n c e s

F 10.7           STARTUP           restart identifier from OB

OB  20:1/AN      OB  20:1/S        OB 21:1/AN        OB  21:1/S
OB  22:1/AN      OB  22:1/S        FB    5:2/AN      FB     5:2/O
FB     5:2/R     FB   10:2/AN      FB  10:2/O        FB   10:2/R
```

Figure 2-5 References to the Operand -STARTUP in CARWAS



The “documentation” menu provides you with a series of lists in which the cross references are compiled either for a single operand (in this case F 10.7) or for a group of operands (e.g. I, Q, F, counters). The cross references can be restricted to a particular block or extended to cover all the blocks in the program. *Figure* 2-6 shows the printout of the cross references for the “outputs” in FB5 and the “counters” and the start-up flag (F 10.7) in all blocks. The asterisks beside segment numbers indicate that the operand occurs in an assignment. You can select the list you require by marking the options in the job box “Output XRF list”.



```
FB 5                         C:  CARWASST.S5D              LIB=2              LEN=166
X reference list: outputs

Q              32.0   -C-FWDS          SEGM.  :          7*,  8*,  9*,  10*,  15*
Q              32.1   -C-BWDS          SEGM.  :          4*,  8*,  9*,  10*,  11*,  15*
Q              32.2   -OPEN-D          SEGM.  :          4*,  12*,  15*
Q              32.3   -CLOSE-D         SEGM.  :          6*,  7*,  15*
Q              32.4   -CAR-IN          SEGM.  :          5*,  6*
Q              32.5   -CAR-OUT         SEGM.  :          4*,  5*,  13*,  14*
Q              32.6   -ROTATE–B        SEGM.  :          7*,  9*
Q              32.7   -SHAMPOO         SEGM.  :          7*,  8*
Q              33.0   -RINSE           SEGM.  :          8*,  9*
Q              33.1   -WAX             SEGM.  :          9*,  10*
Q              33.2   -DRY             SEGM.  :         12*
QB             32   -                  SEGM.  :          3*,  4*
QB             32   -                  SEGM.  :          3*,  4*

X reference list: counters
               FB           5 :    Processed
               FB          10 :    Processed
               OB           1 :    Processed
               OB          20 :    Processed
               OB          21 :    Processed
               OB          22 :    Processed

C               2            -STEP      FB  5       3*,  4*,  5*,  6*,  7*,  8*,  9*,  10*,  11*
                                                   12*,  13*,  14*
                                        FB 10       3*,  4*,  5*,  6*,  7*,  8*,  9*,  10*,  11*
                                                   12*,  13*,  14*
                                        OB 20       1*
                                        OB 21       1*
                                        OB 22       1*
C              20            -NUMBER    FB  5       6*
                                        FB 10       6*

   Search  for an  operand in all blocks

F              10.7          -STARTUP   FB  5       2*
                                        FB 10       2*
F              10.7          -STARTUP   OB 20       1*
                                        OB 21       1*
                                        OB 22       1*
```

Figure 2-6   List of Cross References from the Carwash Program

Search

Editor

STEP 5 blocks >

in the program file...

During the editing session, you can specify cross references to be searched for.

1. Call FB5 in the block editor and press ***F3*** = *Search.*
2. As the search key (KEY:) specify an operand, in this case I 32.4 or -C-FRONT. Press ***F2*** *(From seg1).*

The first occurrence of this operand is displayed in segment 8 statement 4.

3. Press ***F3*** = *Search* again and ***F3*** = *Continue.*

Segment 10 is displayed with the cursor marking statement line 4, etc.

Rewiring

Management
Rewire >
Manual...

It is sometimes necessary to assign an operand a new address within the program. Using the “rewiring” function, operands can be renamed, i.e. assigned different I/O addresses.

To illustrate how this function works, we will rename one of the output operands in FB10.

1. Check the file name:
   Program file C:CARWASST:S5D to
   Program file C:CARWASST:S5D
2. Enter FB10 in the job box and confirm with ***OK***.

A table appears in which you enter the previous operand (in absolute representation) on the right-hand side and the new operand on the left-hand side.

3. Type in the old operand: Q 33.2 new operand: Q 1.7.
4. Complete your input with the ***Insert*** key and confirm the following system messages with ***yes***.

Check that the modification has been made as follows:

5. – Call block FB10 in the editor and press ***F3*** *= Search,*
   – Type in the search key Q 1.7 and press ***F2*** *(From seg 1).*

Segment 12, operand Q 1.7 is entered three times instead of -DRY, i.e. the signal to open and close the air valves for drying the car is now output via Q 1.7.

Comparing Blocks

STEP 5 provides a compare function with which blocks of the same type and same number in the PLC and PG can be compared. If there is no PLC connected, blocks in different programs can be compared with each other. To try out this function, you can compare the FB10 in CARWAS... modified with the rewiring function with the original FB in PROEXA... .

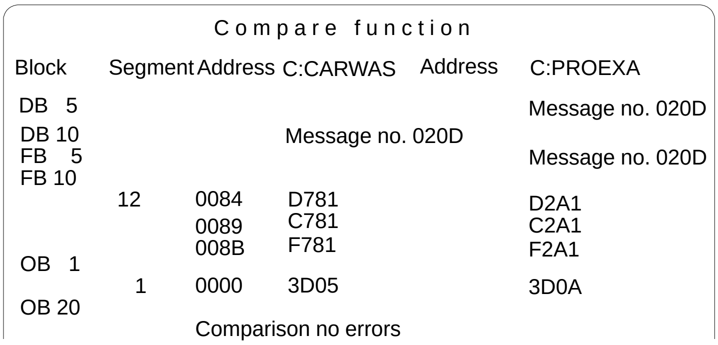
Project
Blocks >
Compare >
File – File ...

1. In the Object menu, call the functions “Blocks, Compare, File - file”.
2. In the job box, enter C:PROEXAST.S5D under “compare with program file” and FB10 under block list. When you have done this, click on ***OK***.

You then obtain an overview of the differences found in segment 12. The differing STEP 5 operations are listed with their addresses in MC5 code.

3. Repeat the block comparison by marking “all blocks” in the job box.

STEP 5 displays the comparisons as shown in *Figure* 2-7. Non-existent blocks are indicated by the message 020D. You can also recognize that different FBs are called in OB1.



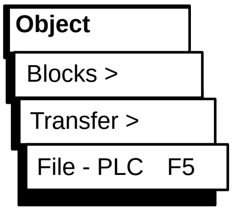
```
                      Compare function
Block    Segment Address C:CARWAS   Address   C:PROEXA
DB   5                                        Message no. 020D
DB 10                    Message no. 020D
FB   5                                        Message no. 020D
FB 10
           12    0084    D781                 D2A1
                 0089    C781                 C2A1
                 008B    F781                 F2A1
OB   1
            1    0000    3D05                 3D0A
OB 20
                 Comparison no errors
```

Figure 2-7 Block Comparison between CARWAS and PROXEA

2.5 Loading and Testing the Program

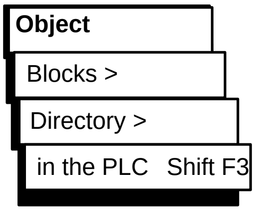
To test the carwash program, you must now connect an S5-95 to your programmer. Establish the permanent connection between the PG and PLC as follows: change the mode to “online[cycl.]” in Page 2 of the “Settings” using ***F3*** = *Select* and ***F6*** = *Save.*

2.5.1 Loading the Program



Complete loading the program using the function “Transfer blocks” in the object menu.

1. Select “Blocks, Transfer” in the object menu.
2. If it is not already set, enter C:CARWASST.S5D as the source in the job box and under selection enter “FB5” in the block list, then “DB5” and finally “all OBs”.
3. After pressing ***Transfer***, the relevant blocks are copied to the PLC. Confirm this with ***OK***.
4. Check the loading by outputting a list of the blocks on the PLC.
5. To do this, once again mark “all blocks” in the job box.



A list of all the blocks loaded on the PLC is output. The list only contains the program sections required by the programmable controller. Comments and block preheaders are not transferred when the blocks are loaded.

Note

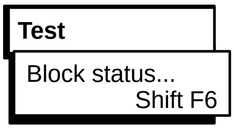
System blocks of the PLC are also output.

2.5.2 Testing the Program

You can now test your user program, i.e. function block FB5, in the online mode segment by segment and statement by statement to make sure that it runs correctly. The decision table (Page A-10) shows you the reactions of the PLC on the output side to certain combinations of input signals.

To set or modify the input signals, you can use the eight on/off switches (I 32.0 ... I 32.7) and two buttons (I 33.0/I 33.1) on the “SIMATIC INPUT” simulator (order no. 6ES5788-8MK11). Depending on the required method of representation of the signal status displays on the PG, select the function “block status” or “status variable” to test the signals.

2.5.3 Block Status



1. On the simulator, switch all the toggle switches down (= off) and set the mode selector on the PLC to STOP.
2. Select “Block status” in the test menu.
3. Enter FB5 in the job box, mark the options with **yes** and click on ***OK***.

Segment 1 appears in the “STL” method of representation. Below the header information, the statement, the result of logic operation RLO and the status of ACCU 1 and ACCU 2 are displayed. The entries in the columns “status” (result condition codes) and “SAC” (address counter) are irrelevant for testing the example.

Now switch the PLC to RUN.

The corresponding RLO is displayed and at the bottom right the message “Status processing active” appears.

1. Start the carwash by flicking up the switches for I 32.0 and I 32.1 (= on).
2. Move the breakpoint for status processing to segment 3 by pressing ↓↓↓ = scroll forward twice.
3. Move the cursor to the line following the jump operation by pressing ***cursor down*** three times.

The displays (RLO, Status etc.) disappear and you can see that this statement (following the branch) is not processed (message “Statement not processed). In segment 4, the situation is similar. The processing also stops at the branch.

4. Now move the breakpoint to segment 5, in which according to *Figure KEIN MERKER* the actual washing process begins.

RLO=1 in line 1 indicates that all the prerequirements such as the initial carwash position and the step counter (-STEP) setting have been fulfilled and the washing process can begin. In column 5 of *Table* A-2 you can see which inputs must be set.

5. Flick the switches E32.5 and E32.6 up.

The step counter and ACCU 1 have the value 1, the set inputs have the status 1. On the PLC, output Q 32.4 is lit, i.e. DRIVE CAR IN is displayed.

6. Move the breakpoint to segment 6 and flick I 32.3 up for “car in position”. After pressing the button I 33.0 (start) the washing process is started.

The display for the driver goes off (Q 32.4 = 0) and the door is closed (Q 32.3 is lit). The step counter (-STEP) changes to 2.

7. Move the breakpoint to segment 7 and simulate the closed door by I 32.6 = off and I 32.7 = on.

The parts of the process “apply shampoo”, “rotate brushes” and “carriage forwards” are started (variable = 1). The step counter switches to 3.

8. Simulate the remaining parts of the washing process by changing the inputs according to *Table* A-2 depending on the position of the breakpoint.

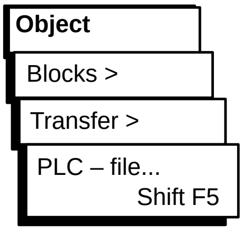
In segment 11, following I 32.5 = 1, you can see how the wax distribution time WT is decremented to 0 at one second intervals followed by the start condition for drying being generated automatically by the step counter (= 7).

9. Move the breakpoint to segment 12.

You can follow the drying time (DT = 45 s). Simulate the remaining parts of the process in step 8 and step 9 as described above.

In segment 14, the step counter returns to 1, indicating the initial position of the carwash. This means that the example program is capable of running and fulfilling the task. If errors occur, they must be corrected using the information provided by the RLO and contents of the ACCUs and the status of the signals.

1. Change to the editing mode with ***F6***. You can position the cursor on the statements you want to modify, delete or insert.
2. Press the ***Insert*** key and answer the prompt “Enter modified segment?” and the next message with ***yes***.

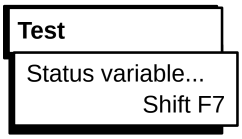


With the steps outlined above, you have modified the program in the PLC. To transfer the modified block to the PG, e.g. for archiving,

3. Select “Blocks, Transfer, PLC - file” in the object menu and enter FB5 in the job box.

Status Variable

1. Set all the toggle switches on the simulator to off and the mode selector on the PLC to STOP.
2. Select “Status variable” in the test menu.

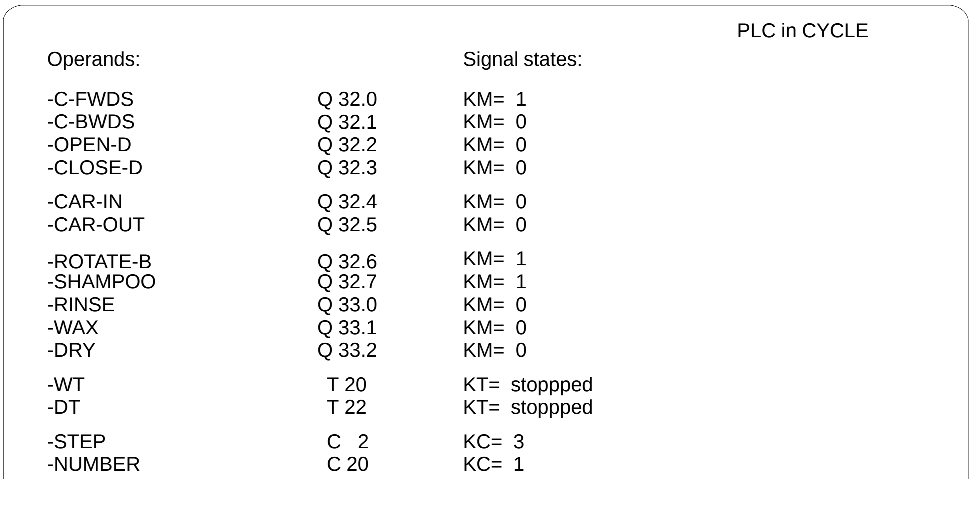


An empty table with the columns “Operands:” and “Formats:” appears on the screen.

3. Working from the signal list, enter all the output operands including timers and counters in absolute or symbolic format and complete each line with the ***Return*** key.

STEP 5 adds the format to your entries. Once your operand list has the same contents as shown in *Figure* 2-8

4. Press ***F2*** = *Store* and specify the number of the variables block (in this case VB 5).



```
                                                                  PLC in CYCLE
Operands:                                     Signal states:

-C-FWDS                 Q 32.0                KM=  1
-C-BWDS                 Q 32.1                KM=  0
-OPEN-D                 Q 32.2                KM=  0
-CLOSE-D                Q 32.3                KM=  0

-CAR-IN                 Q 32.4                KM=  0
-CAR-OUT                Q 32.5                KM=  0

-ROTATE-B               Q 32.6                KM=  1
-SHAMPOO                Q 32.7                KM=  1
-RINSE                  Q 33.0                KM=  0
-WAX                    Q 33.1                KM=  0
-DRY                    Q 33.2                KM=  0

-WT                      T 20                 KT=  stoppped
-DT                      T 22                 KT=  stoppped

-STEP                    C   2                KC=  3
-NUMBER                  C 20                 KC=  1
```

Figure 2-8   Display of the Output Operands in Variables Block VB 5

5. Complete these preparations by pressing the ***Insert*** key or ***F6*** *= Activate.*

Test the function block using the “status variable” function as follows:

6. Switch the PLC to RUN and the toggle switches I 32.0 and I 32.1 to “on”.

The current values of the operands (initially all 0) and the messages “PLC in CYCLE” and “Status processing active” are added to the “Signal states” column. By using the decision table, you can once again check the reaction of the controller to certain combinations of values at the inputs.

7. Switch I 32.5 and I 32.6 to “on”.

The carwash goes to the ready status with Q 32.4 = 1 and C 2 = 1.

8. Simulate the car being driven in by I 32.3 = on and starting the carwash by setting I 33.0.

To door is closed (Q 32.3 = 1), the step counter changes to 2 and the action itself is stored in C 20 = 1.

9. Simulate the status “door closed” by I 32.6 = off and I 32.7 = on.

The PG now displays the signal states shown in *Figure* 2-8. The brush carriage now moves forwards with the brushes rotating and the shampoo jets open.

10. Simulate the movement of the carriage “carriage front” or “carriage back” by switching I 32.4 and I 32.5 on and off.

Continue simulating the inputs until the two times WT and DT are displayed and terminated with step counter = 8.

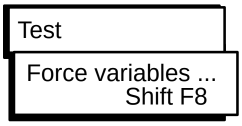
In step 9 (I 32.7 = off, I 32.6 = on) DRIVE CAR OUT is displayed and in the last step (I 32.3 = off) the ready status is re-established with the display DRIVE CAR IN and step counter = 1.

11. To terminate the status function, press ***ESC*** = *Exit* and you return to the menu with ***F8***.

STEP 5 displays the signal statuses at the selected breakpoint. By pressing ***ESC*** = Exit once, you can interrupt the status processing and insert additional operands in the list. Following this, the ***Insert*** key continues the status processing.

Force Variables

With this function you can modify variables (e.g. I/Q/F) in the process image byte by byte. You can also display the current signal states with the PLC in the RUN mode. Once again, an operand list must be prepared for this function.



Test

Force variables ...
Shift F8

Select “Force variables” in the test menu and type in the inputs and outputs as byte operands (IB and QB) in the empty table “Operands - Formats”. Complete each line with the ***Return*** key and overwrite the default format with “KM”. Add C 2/C 20 and T 20/T 22 to the list and then press ***F6*** = *Activate.*

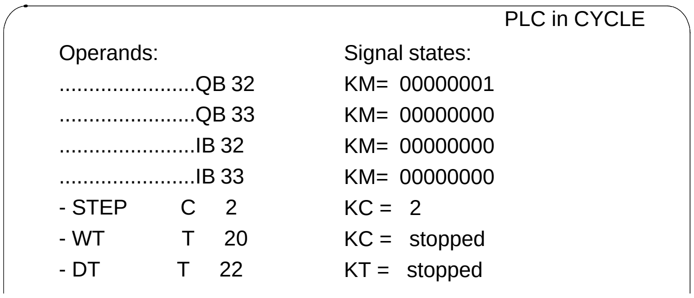
Your screen will then resemble the screen illustrated in the figure below. By activating the switches on the simulator one after the other, you can display the corresponding values at the outputs and counters (much the same as in the status functions).

Press ***ESC*** = *Exit* and switch I 32.0/I 32.1 to “on” and the PLC from STOP to RUN.

The PG now displays the column “Force process image”. You can now influence the outputs in QB 32/QB 33 directly with the keyboard and check the way in which the actuators function. Try this out as follows:

Enter the bit pattern KM = 00110011 in QB 32 and press ***Insert****.*

In the PLC, the output relays 32.0/32.1 and 32.4/32.5 must be switched on and the message “End of force fct.” must appear on the screen.



```
                                              PLC in CYCLE
Operands:                      Signal states:
.......................QB 32   KM=  00000001
.......................QB 33   KM=  00000000
.......................IB 32   KM=  00000000
.......................IB 33   KM=  00000000
- STEP        C     2          KC =   2
- WT          T     20         KC =   stopped
- DT          T     22         KT =   stopped
```

Overview of the Functions 3

This section contains descriptions of all the operations and functions you can use when working with STEP 5. The sequence and contents of the individual sections are oriented on the sequence in which the functions appear in the main and submenus (Overview pages 3-2/3-3).

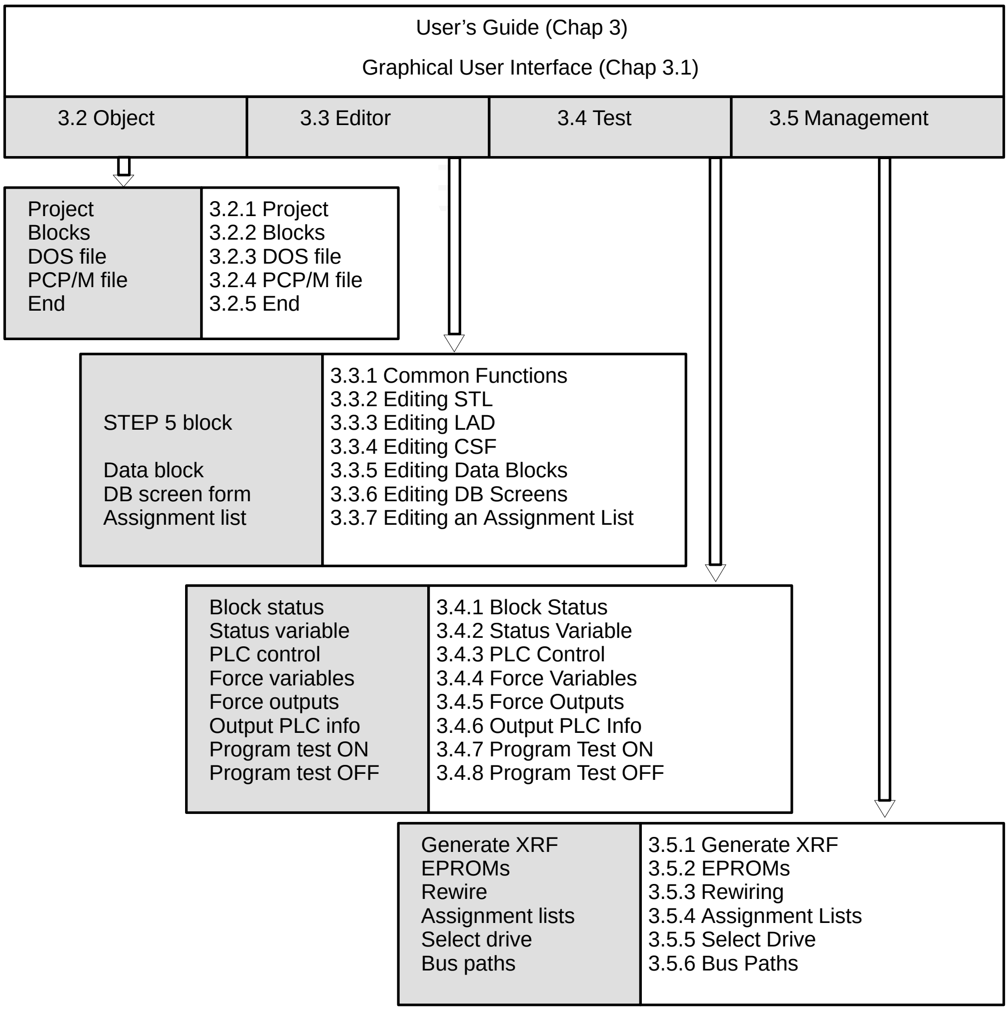


Figure 3-1 Overview: Main Menu → STEP 5 Function → User’s Guide

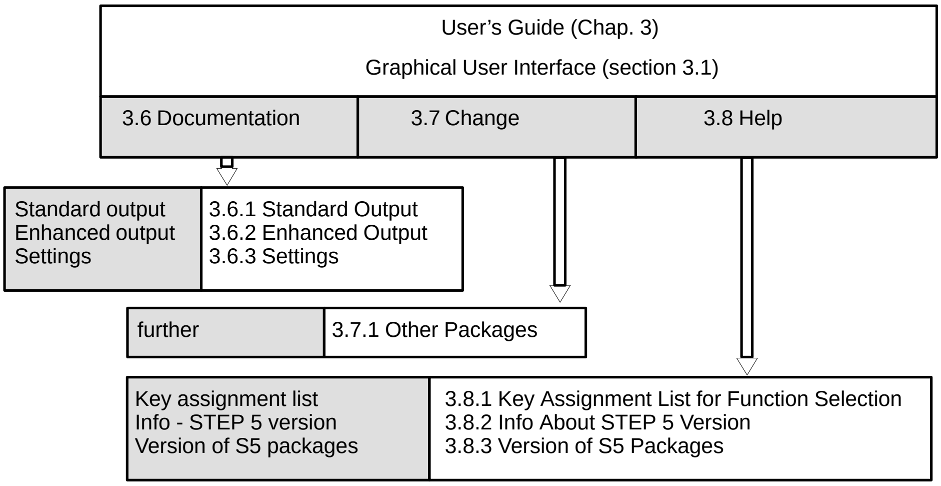


Figure 3-2 Overview: Main Menu → STEP 5 Function → User's Guide (continued)

Key

Menus on the screen are shown with gray shading, the titles of the sections in the User's Guide are shown in italics.

Creating a Program

STEP 5 provides functions and utilities to help you create your programs and for user-friendly

- editing,
- transferring (copying) and
- documentation

of blocks and files.

Online Functions

For testing and correcting your programs and for operating the progammable controller (PLC), STEP 5 has functions

- for controlling and monitoring the PLC (STOP/RUN, info),
- for outputting/displaying the current status of process variables (in the block or in a selectable list),
- for displaying the process image of the peripheral I/Os and for setting/modifying I/O values.

EPROM/EEPROM Program Memory

If the PG has an EPROM slot, you can write (blow) and read (copy) programs to and from the EPROM/EEPROM submodules using STEP 5.

Bus Connection

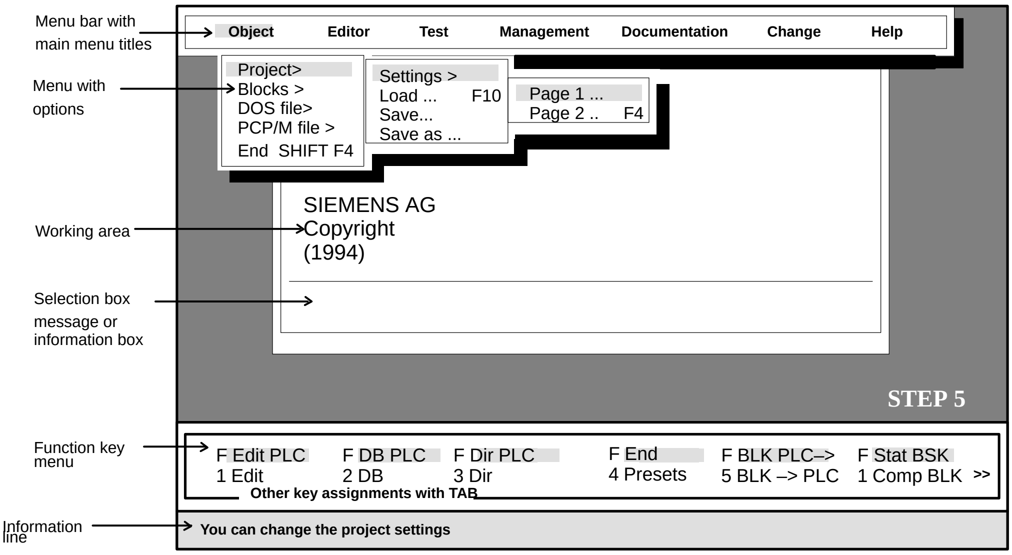
If the PG is connected to a bus, you can establish (edit) bus paths and store/call up the data you have created (path name, addresses, nodes etc.) using STEP 5 utilities.

Operating System

STEP 5 programs and files you created under PCP/M-86 can also be processed under S5-DOS.

3.1 Graphical User Interface

STEP 5 functions are activated using the menu bar with its main menus and submenus. With either the mouse or keyboard, you can select the tools and utilities you require for your session. If you prefer to continue using the function keys as in previous STEP 5 versions, you can, of course, do so.



SIEMENS AG
Copyright
(1994)

You can change the project settings

Figure 3-3 Graphical User Interface

Menu Bar, Menus

When you select a menu item in the menu bar either by clicking it with the mouse or by positioning the cursor on it and activating it with the ***Return*** key, you open the menu. This menu contains options or functions related to the main item. If you select menu items with an arrow (>) to the right of them, you open a further submenu.

Working Area

The selection boxes in which you make settings, the information and message boxes and the windows of the program editors are displayed in the working area of the screen.

S5 Identifier

This displays the package you are currently working with, for example, STEP 5 or another S5 package such as GRAPH 5.

Function Key Menu

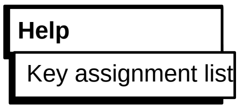
The function key menu allows you to call certain selection boxes or editors directly without a longer series of selections or keystrokes.
To display the remaining functions assigned to function keys simply press the ***TAB*** key or click the symbol “>>” to the extreme right of the display.
You can trigger functions provided by the function key menu in the following ways:

- Click the field containing the name of the function using the mouse.
- The functions in the lower row can be activated by pressing the function key with the number shown to the left (***F1*** to ***F10***).
  You activate the functions displayed in the top row on a shaded background by holding down the ***SHIFT*** key and pressing the function key with the number displayed to the left of the field (***SHIFT F1*** to ***SHIFT F9***).

Info Line

The information line provides information about the menu item you have selected (submenu or menu function) but not yet activated.

Help



Key assignment: you can obtain more detailed information about the functions assigned to the keyboard by activating the key assignment list function in the help menu. When selection boxes are displayed which have their own function key assignment, you can obtain information about the function keys by pressing ***SHIFT F8***. If the prompt ***Continue? yes/no*** is displayed, you can obtain more information about the individual functions by selecting ***yes***.

Information Line

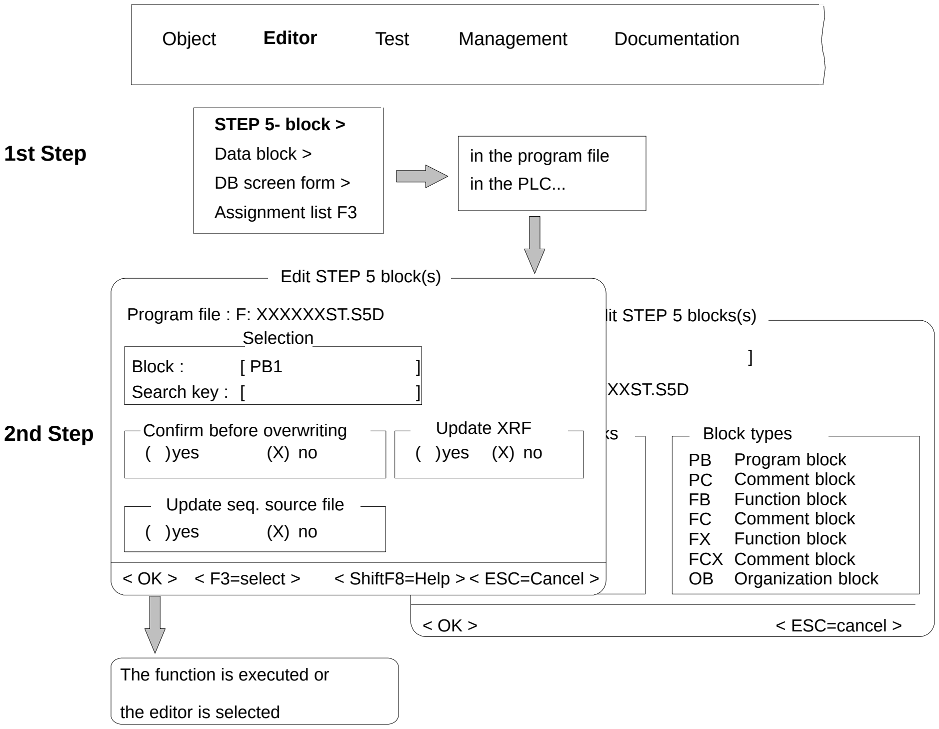
The information line provides you with information about the menu item (submenu title or function) currently selected but not yet activated.

Function Selection

You call a function or an editor in two steps, as follows:

1. Select the function in the main or submenu.
2. Specify the function by entering parameters in the job box and confirming your input.

The function is then started/executed or the editor is called.



Program file : F: XXXXXXST.S5D

Selection

Block : [PB1]

Search key : [ ]

Confirm before overwriting: ( )yes (X) no

Update XRF: ( )yes (X) no

Update seq. source file: ( )yes (X) no

Block types

PB	Program block
PC	Comment block
FB	Function block
FC	Comment block
FX	Function block
FCX	Comment block
OB	Organization block

The function is executed or the editor is selected

Figure 3-4 Selection of the Main and Submenus

3.1.1 Function Selection

Function Selection

With the main menu displayed, you can select a function or an editor using the appropriate submenu. Depending on the current operating status of the PG, e.g. when there is no connection to a PLC, certain functions cannot be activated and do not respond in the menu. There are four ways of selecting a function, as follows:

Mouse

1. Click on the required function with the mouse.

Cursor Keys

2. Move through the main menu boxes using the ***cursor right, .. left*** keys and in the submenus using the ***cursor up .. down*** keys. You display further submenu levels or call selected functions with the ***Return*** key. You can clear displayed submenus from the screen using ***ESC*** = *Exit*, if necessary several times.

Hotkeys

3. You can press the letter displayed in red in the function name (on monochrome screens this letter has a black background).

Function Keys (Fast Selection)

4. You can press a function key to select a regularly required function directly (→ *Graphical user interface, Function keys).*

Example

Calling the block editor.

Mouse

Click on the following menu options one after the other

1. ”Editor” in the main menu
2. ”STEP 5 block” in the submenu,
3. ”in the program file” in the following submenu

The “Edit STEP 5 block(s)” job box is displayed.

Cursor keys

with the cursor keys:

1. Select “Editor” with the ***cursor right/left*** keys in the main menu and press the ***Return*** key.

The editor submenu is opened.

2. Press the ***Return*** key.

The submenu “STEP 5 block” is opened.

3. Select the function “in the program file” with the ***cursor up/down*** keys and press the ***Return*** key.

The “Edit blocks” job box is displayed.

Hotkeys

Starting from the main menu, press the following keys (upper/ lower case irrelevant):

1. ***E*** for “Editor”
2. ***S*** for “STEP 5 block”
3. ***I*** for “in the program file”

The “Edit STEP 5 block(s)” selection box is displayed.

Function Keys

Press ***F1*** in the row of function keys on the keyboard.

The “edit blocks” selection box is displayed.

Key to the operating menus

The function names in the menus have been supplemented by characters and markers with the following significance:

Red letter or with black background	=	The function can be activated (by pressing this character)
Followed by “>”	=	There is a further submenu to follow
Followed by “. . .”	=	There is a job box to follow
Function with no following character	=	Selecting this menu box activates the function immediately.
F”n” or ***SHIFT F”n”***	=	Indicates that fast selection of the function is possible using the function key indicated.

Job/Selection Box

If more information is required before a function can be executed (marked by "..." in the menu), STEP 5 opens a job box (*Figure 3-5*). In some situations, this box is adapted to the individual functions and can have more or less input fields. In these input fields, you enter the name or the parameters to specify your task or job. Displayed objects and options are selected by entering an "x". When you complete the name fields, you can display information about existing or possible blocks (block types) in the selection box and select them directly.

When you fill in the name fields, you can display information about existing blocks or possible block types or files in the selection box by pressing ***F3*** and then select them in the box.

Logically incompatible parameters are rejected by the system and a self-explanatory message is displayed in red or with a black background in the job box.

There are a variety of boxes, each responsible for a certain task:

– Job boxes as shown in *Figure* 3-5. The corresponding function can only be activated from this box.
– Block selection boxes as shown in *Figure* 3-6.
– File selection boxes as shown in *Figure* 3-7.
– Job boxes of varying sizes where further inputs can be made and which are adapted to the requirements of the particular function.

Positioning the cursor and making entries is the same for all job, block selection and file selection boxes.

Note

Input fields and particular features which are only important for certain functions are not described here but are dealt with in the description of the function.

Job box

This box type (*Figure* 3-5) is always displayed when you want to activate a function within a function menu.

Program File (1)

With all functions specific to the program, the name of a STEP 5 program file from the “Settings” (→ *Project*) is entered as a default by STEP 5. Depending on the requirements for the individual functions, this field is either blocked or an entry can be made.

Selecting the Objects (2)

You make your selection by marking (X) the displayed options or writing in the square brackets as follows:

- a block (name)
- a block list (single blocks separated by commas)
- a file name
- a search key
- a number depending on the particular input field (e.g. from [ ] to [ ])

Outputting the Objects

Depending on the particular function, you can output the objects defined in the box as follows:

- on a printer
- on the screen
- to a file. The name of the file is taken from the “Settings” box, however, you can change it there.

Printout Type

You can select the layout of the output to suit your purpose or set the LS.INI file.

Output Options

Depending on the particular function, you can select the following:

- a block and corresponding block header
- FB with names.

Selecting Input Fields

Fields in the job box and key functions (→ *Appendix 4 Key Assignment*) that can be used to guide the cursor in the fields of the job box (*Figure* 3-5).

(2) Selection field

Selection boxes for blocks or files are displayed by pressing ***F3***. These fields have a colored background or are displayed inversely.

(3) Fields within a frame

These fields contain specific selections, options and destinations for output. This is indicated by a frame around the field. If you do not have a mouse, you can select these fields with the ***TAB*** key. Note that in “Output on/to”, the field “Name” is selected with this key.

(4) Input fields

Within the fields in frames, you can select the input fields using the ***cursor*** keys. Either a name or an X is entered at the position marked by the cursor.

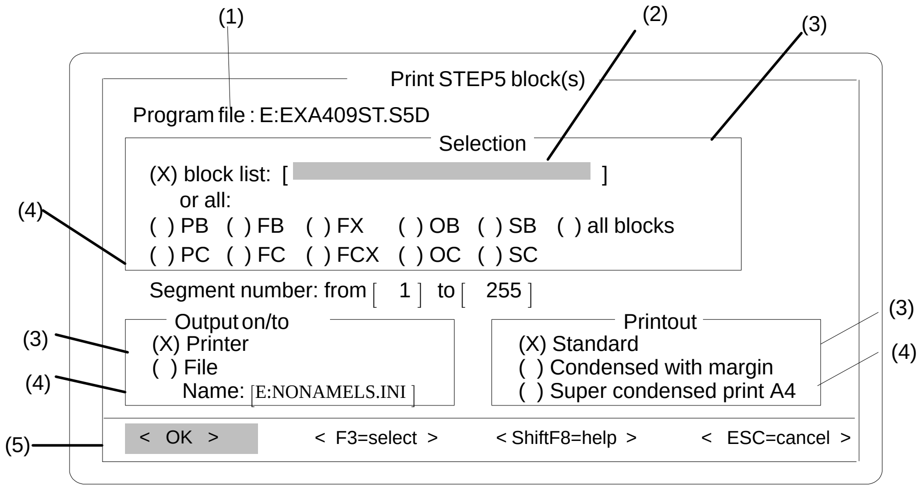
(5) Commands

OK = *Return* The selected parameters are entered and the function activated.

ESC = *Cancel* Cancels without entering parameters.

SHIFT-F8 = *Help* A help text is displayed for the field marked by the cursor

F3 = *Select* Selection box for calling blocks or files.



Print STEP5 block(s)

Program file : E:EXA409ST.S5D

Selection

(X) block list: [ ]
or all:
( ) PB ( ) FB ( ) FX ( ) OB ( ) SB ( ) all blocks
( ) PC ( ) FC ( ) FCX ( ) OC ( ) SC

Segment number: from [1] to [255]

Output on/to
(X) Printer
( ) File
Name: [E:NONAMELS.INI]

Printout
(X) Standard
( ) Condensed with margin
( ) Super condensed print A4

Figure 3-5 Example of a Job Box

Block Selection Box, File Selection Box

If the cursor is positioned in a name field (*Figure* 3-5) and you press the **F3** key, a block or file selection box is displayed. In these boxes (*Figure* 3-6/*Figure* 3-7) you can select an object directly for processing.

(1) The blocks/files in the set program file are listed in this field. If the cursor is positioned on one of these objects and you press the ***Return*** key, the object is entered in the “blocks” or “file name” field and is therefore selected for processing. If you require several blocks for processing, these must be entered singly in the “blocks” field. If you click on ***OK*** or press the ***Return*** key, the box is closed and the name of the selected object is entered in the selection field of the job box.

(2) Here, you can change the following:

Block type — You can select a type with the ***cursor*** keys or can double click on a type with the mouse. All blocks of this type are displayed in the field (1).

Drive — You can select the drive with the ***Return*** key or by clicking on a drive with the mouse. The directories in this drive are displayed in the field (2).

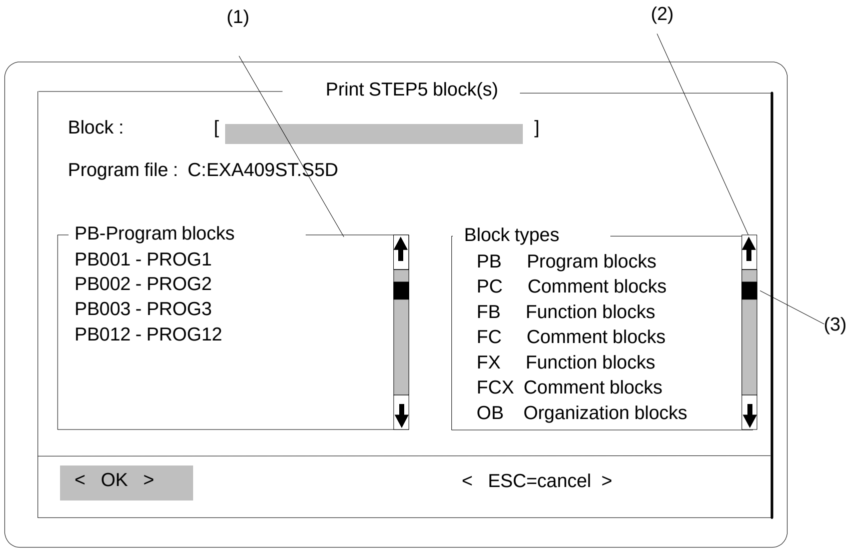
Directory — You can select the content of a directory by pressing the ***Return*** key or double clicking a directory. The files in the directory are displayed in the field (1).

(3)
(5) Using the ***roll screen*** (up or down) keys or by clicking on the bar of the field, you can move the contents of the field up and down. The cursor keys move the list one line up or down.

(4) The search key entered here is searched for in the selected directory. If it exists, the file name is displayed in the field (1).

Note

Browsing in drives and directories is not permitted in all file selection boxes.
User files must not be located in directories with system files (S5*.CMD and S5*.DAT).



Print STEP5 block(s)

Block : [ ]

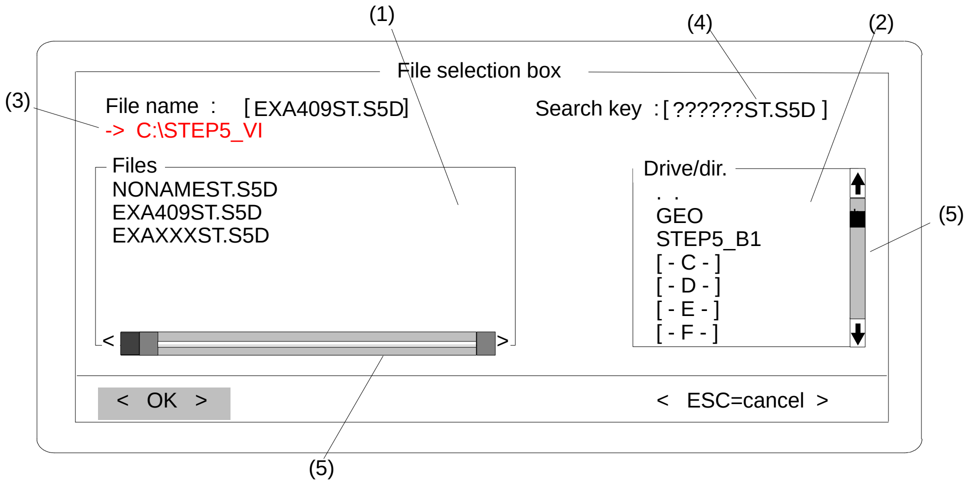
Program file : C:EXA409ST.S5D

PB-Program blocks
PB001 - PROG1
PB002 - PROG2
PB003 - PROG3
PB012 - PROG12

Block types

PB	Program blocks
PC	Comment blocks
FB	Function blocks
FC	Comment blocks
FX	Function blocks
FCX	Comment blocks
OB	Organization blocks

Figure 3-6 Example of a Block Selection Box



File selection box

File name : [EXA409ST.S5D]
-> C:\STEP5_VI

Search key : [??????ST.S5D]

Files
NONAMEST.S5D
EXA409ST.S5D
EXAXXXST.S5D

Drive/dir.
. .
GEO
STEP5_B1
[- C -]
[- D -]
[- E -]
[- F -]

Figure 3-7 Example of a File Selection Box

Tip

If part of the screen display is covered by a message box or a prompt such as “Continue? yes/no”, you can toggle the display with the space bar.

Function Keys

Function keys are keys which are assigned a function by the software and which are used to activate software tasks using the keyboard or using buttons on the screen.

To allow STEP 5 to be operated to suit the situation or task in hand, the function keys are available at various levels and can be used to trigger the displayed function.

F1...F10
SHIFT F1...F9
(hardkeys)

1. Function keys can be used as shortcut keys to select STEP 5 functions from the main menu of the graphical user interface. You can display the functions of the keys by selecting “Key assignment list” in the Help menu. When you open a menu or submenu, you will see that some functions have a function assigned to them. In the editors and dialog boxes, you can display the function key assignment by pressing ***SHIFT F8***.

OK, ESC, yes, no
(box buttons)

2. The STEP 5 selection, information and message boxes all have a line containing buttons at the bottom edge of the box. Using these buttons you can confirm, discard or activate operations.

F1...F8
SHIFT F1...F8
(function keys)

3. In the editors, STEP 5 normally displays a line of 8 (or 16) function keys at the bottom edge of the screen which can be used to activate editor functions.

Space Bar

You can show or hide prompts and messages with the space bar. This allows you to see the parts of the screen behind messages.

ESC Key

You can exit dialog boxes with only one button by pressing ESC.

3.2 Object

In this menu you can select the following functions:

→ ***Project***
You select all the settings for a project with the corresponding safeguards in this menu. Once you have saved a project, you can load it from this menu.

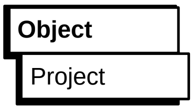
→ ***Blocks***
Copying blocks and getting information about blocks.

→ ***DOS files***
Copying files and getting information about the files.

→ ***PCP/M file***
Copying files and getting information about PCP/M files.

→ ***End***
Exits STEP 5.

3.2.1 Project



Before you begin to program with STEP 5, you should plan the following information:

- some or all the required file names of the user program,
- a working directory containing all the files.
- project-specific parameters such as type of representation or mode,

You only need to make these settings once with STEP 5. Specifying a unique directory which will contain the files belonging to one project makes it far easier to organize your programming. STEP 5 saves all these settings in a project file (*PJ.INI) of which you can make copies. With this, you have a schedule of all the relevant data for a project.

You can change the settings at any time to match them to new conditions. Once you load a project file, the data are available immediately and you can begin programming without having to create new settings.
The files themselves must, of course, be in the directory selected in the project file.

Figure 3-8 shows how the project file and corresponding program files are organized. The project file is in the same working directory as the files. The settings in the project file relate to these files. Exceptions to this are the printer file and the path file. These are always in the directory in which STEP 5 is loaded.

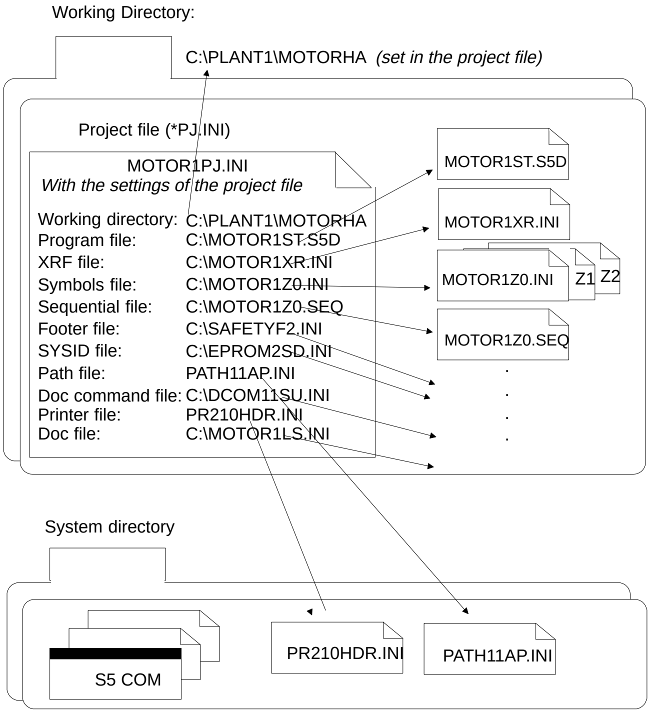

Working Directory:
C:\PLANT1\MOTORHA *(set in the project file)*
Project file (*PJ.INI)
MOTOR1PJ.INI
With the settings of the project file
Working directory: C:\PLANT1\MOTORHA
Program file: C:\MOTOR1ST.S5D
XRF file: C:\MOTOR1XR.INI
Symbols file: C:\MOTOR1Z0.INI
Sequential file: C:\MOTOR1Z0.SEQ
Footer file: C:\SAFETYF2.INI
SYSID file: C:\EPROM2SD.INI
Path file: PATH11AP.INI
Doc command file: C:\DCOM11SU.INI
Printer file: PR210HDR.INI
Doc file: C:\MOTOR1LS.INI
MOTOR1ST.S5D
MOTOR1XR.INI
MOTOR1Z0.INI
Z1
Z2
MOTOR1Z0.SEQ
System directory
S5 COM
PR210HDR.INI
PATH11AP.INI

Figure 3-8 Organization of the Project File and Program File

The following functions are available in the “Project” menu.

- **Setting** all the parameters required for a specific project (→ *Settings*). Boxes are available in which you can set the following:
  - Files belonging to a project. These files are always set in the job and selection boxes or editors in which they will be addressed.
  - Parameters, e.g. symbols, method of representation (LAD, CSF, STL) etc.

Once you have selected these settings for a project, you can only process this project.

- **Loading the settings for a project** (→ *Load*). All the settings for the selected project are loaded. Once the project is loaded, only the files belonging to this project are made available for processing.
- **Saving the project setting** (→ *Save*). All the settings are saved in the file specific to the project.
- **Saving under a new project name** (→ *Save as*). All the settings are saved in a selectable file for the specific project.

Settings

Object
Project
Settings

Before beginning with the actual programming, you select all the parameters required for a project in the displayed setting box. This box is divided into two selectable pages. The parameters you set (e.g. file names) are then later entered in the corresponding job or selection boxes automatically. These settings are brought together in a project file. Selected files and parameters are always valid for the whole project during a sitting.

Note

The settings are retained even when you terminate STEP 5. When you start the next session, the last settings are adopted.

These settings are only saved in a *PJ.INI file when you save them explicitly (→ *Save*). You can load saved settings at any time (→ *Load*).

In general, one part of the names of system files is fixed (e.g. ***Z0.INI**), and one part has 1 to 6 characters that you can select. For example, the symbols file EXA409**Z0.INI** consists of the fixed part in heavy print and the name “EXA409”.

Operation

You can move to the input files within the settings box (fields **1, 2, 3**) either using the ***cursor*** keys or with the ***TAB*** key. For pages 1 and 2 of the “settings”, there are softkeys at the bottom of the settings box. These softkeys are explained below.

Key	Function
F3	The cursor must be located on the label of an input field (field **1** or **2** *Figure* 3-9): 1. You can change parameter settings (e.g. YES/NO or RW/PROT) with ***F3***. 2. Character input is activated. The cursor flashes on the input file (field **2**). Confirm your entries with the ***Return*** key. 3. If you press ***F3*** again, a file selection box is displayed. You can navigate through this box and select a file (→ *Job box*).
F4	Toggle between the pages.
F6	The selected parameters are saved in the *PJ.INI file displayed in field **4**.
Shift F6	With the file selection box open, you can use this key combination to select an existing PJ.INI file.
F7	A help text is displayed for the field marked by the cursor.
F8	Returns you to the previous level. Settings you have changed are entered in the *PJ.INI file.

Page 1

Object

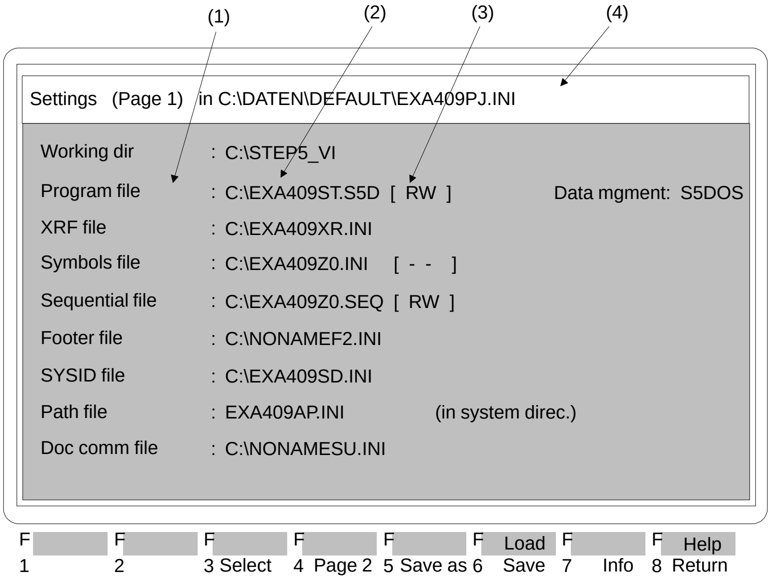
Project

Settings

Page 1

The box “Settings (page 1)” is displayed. Here, you set the files for a specific project.

Example



Settings (Page 1) in C:\DATEN\DEFAULT\EXA409PJ.INI

```
Working dir      : C:\STEP5_VI
Program file     : C:\EXA409ST.S5D  [  RW  ]          Data mgment:  S5DOS
XRF file         : C:\EXA409XR.INI
Symbols file     : C:\EXA409Z0.INI   [  -  -   ]
Sequential file  : C:\EXA409Z0.SEQ  [  RW  ]
Footer file      : C:\NONAMEF2.INI
SYSID file       : C:\EXA409SD.INI
Path file        : EXA409AP.INI              (in system direc.)
Doc comm file    : C:\NONAMESU.INI
```

Figure 3-9   Example of Page 1 of the “Settings” Box

Input Field	Explanation
Working directories	A working directory consists of the drive and a directory (e.g. C:\S5_DATEN\EXAMPLE). This entry represents the complete path under which the files you have entered or that are displayed are stored. You must already have created the directory under MS-DOS. If you have working directories on different drives, it is possible to specify a separate working directory for every file specified in the settings box (except for printer and path files). When you create a working directory containing the system files (S5*.CMD and S5*.DAT) a warning is displayed since user files should not be kept in the same directories as system files.
Program file	Here, you enter the drive and a name up to 6 characters long. The file is then automatically assigned to the working directory and saved in it. If you select the name of an existing program file and if there is no current cross-reference list (XRF file), a box appears in which you can generate a current cross-reference list immediately. **Drive names only up to J** 1. If you do not enter a name, the last name entered is used automatically. 2. If you enter less than 6 characters, the name is padded out with the @ character. Selectable file mode (*field **3**, Figure* 3-9): RW: Read, write possible PROT: Reserves exclusive access rights to the file. Access by other S5 systems is no longer possible. File mode set by STEP 5: RESD: The file is currently reserved. A different S5 system is accessing the file. Once the access is complete, this entry is cleared. RO: Read only.

Input field	Explanation
XRF file	The name of the file (*XR.INI), which will contain the cross reference list, is only displayed here and cannot be modified. For creating the XRF file, refer to (→ *Management, Generate XRF*)
Symbols file	The name of the symbols file (*Z0.INI). If you set this file, then providing you have set “Settings/Page2/Symbols”, you work with symbolic operands (in the editors and in documentation output). As soon as you set this file, the setting for the sequential file is made automatically. The file mode for the symbols file can be selected (refer to program file).
Sequential file	The source file (*Z0.SEQ) which contains the assignment list , is set as soon as you have named the symbols file. The file mode can be selected (refer to program file).
Footer file	The name of the footer file. This footer is automatically output for → *Documentation*. Depending on your selection for the **footer** parameter (Settings/page 2) either **F1.INI** for an 80 character wide footer or **F2.INI** for a 132 character wide footer is entered.
SYSID file	This contains the system identification. It is required for blowing EPROMs and editing a bus path. The file is created with other S5 systems. It is advisable to load this file in the selected working directory. You can, however, specify the file name later in the appropriate dialog box.
Path file	The path file (*AP.INI) contains the bus paths stored under a path name. You can also specify these later in the dialog box for editing a bus path. The path file is always stored in the **system directory** (in which STEP 5 is loaded).
Doc command file	The name of the file in which doc commands will be stored.

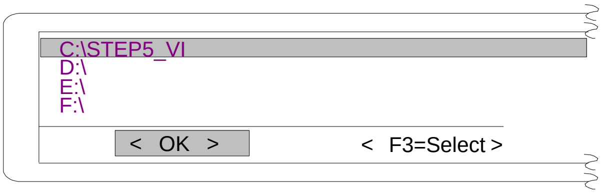
Example of Setting a Working Directory

STEP 5 supports you when setting up the working directory. How to make the best of these functions is explained below.

Ready to Start?

The box “Settings (page 1)” must be displayed.

1. *Position the cursor on the “Working directory” field and press* ***F3****.*



C:\STEP5_VI
D:\
E:\
F:\

Figure 3-10 Example of a Box with Directory Entry

A box is displayed with the working directory selected in “Settings (page 1)”.

2. *Position the cursor on the drive (here C: \) and press* ***F3*** *= Select.*

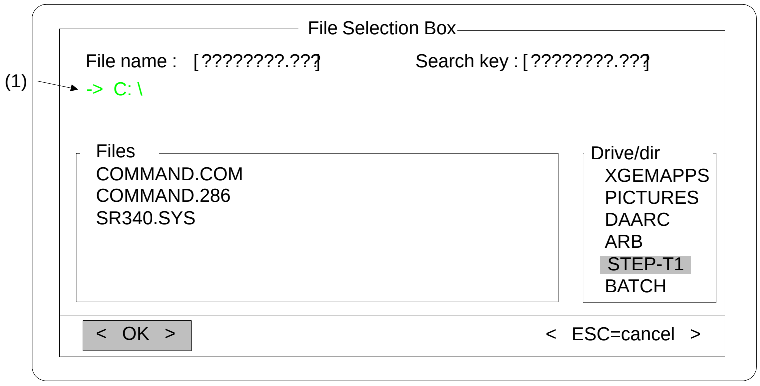
   The “File selection box” is displayed (*Figure* 3-11) with the selected directory C:\ entered in field **1**.

3. *Using the* ***TAB*** *key, move the cursor to the “DR/directory” field and position the cursor on the required directory (here, STEP5-VI*).

4. *Complete your selection with the* ***Return*** *key.*
   The selected directory is now entered in field **1**.
   The “Files” field displays all the files in this directory. These are only displayed for information and cannot be selected.

5. *Press the* ***Return*** *key again.*
   The box displaying the directory appears (*Figure* 3-10). The directory setting displayed previously in field (**1**) (*Figure* 3-11) is adopted.

6. *Press the* ***Return*** *key (****OK****).*
   The “settings page 1” box is displayed again and the working directory is entered in it.



File name : [????????.???]

Search key : [????????.???]

-> C: \

```
COMMAND.COM
COMMAND.286
SR340.SYS
```

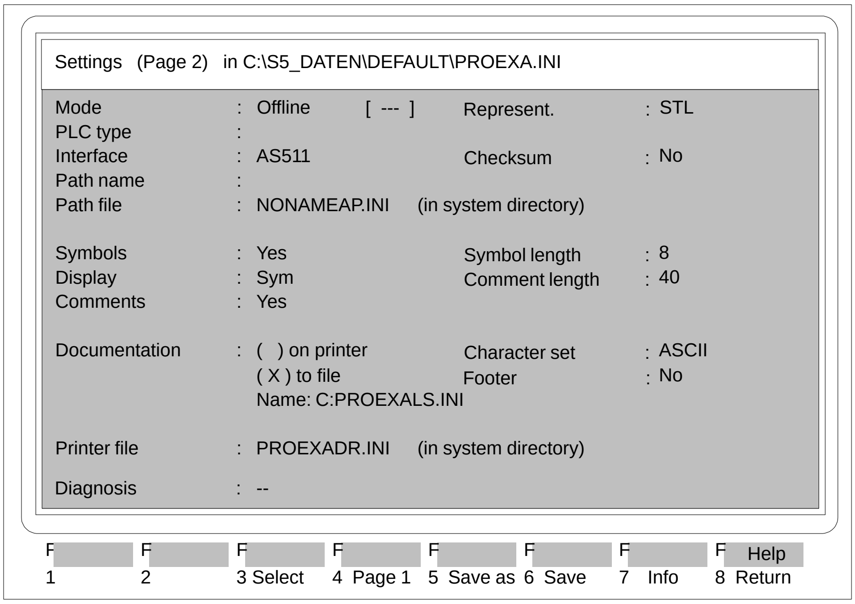
```
XGEMAPPS
PICTURES
DAARC
ARB
STEP-T1
BATCH
```

Figure 3-11 Display of Existing Directories and Files on Drive C:\

Page 2

Object
Project
Settings
Page 2

The “settings” box page 2 is displayed. Here, you select data for the specific project. You can move from field to field as in page 1 using the cursor.



```
Settings   (Page 2)   in C:\S5_DATEN\DEFAULT\PROEXA.INI

Mode            :  Offline      [ --- ]   Represent.        :  STL
PLC type        :
Interface       :  AS511                  Checksum          :  No
Path name       :
Path file       :  NONAMEAP.INI    (in system directory)

Symbols         :  Yes                    Symbol length     :  8
Display         :  Sym                    Comment length    :  40
Comments        :  Yes

Documentation   :  (   ) on printer       Character set     :  ASCII
                   ( X ) to file          Footer            :  No
                   Name: C:PROEXALS.INI

Printer file    :  PROEXADR.INI    (in system directory)

Diagnosis       :  --
```

Figure 3-12 Example of Page 2 of the “Settings” Box

Operation

The following overview describes the possible inputs and parameters you can select in page 2 for a specific project.

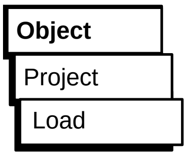
Input field	**Explanation**
Mode	
Offline	No connection to the PLC.
Online	Establishment of a permanent connection to the PLC. The user programs (blocks) can be processed in the PLC via the physical and logical connection: • If a path name is set, the connection is via the bus path. • If no path name is set, the connection is direct. The establishment of the connection is checked. If no connection can be set up, the message “PLC timeout” appears. If the PG-PLC connection is interrupted, the PG is only operational again when the monitoring time set has run out.
Dynamic	This mode is only possible when there is a connection via a bus path. The connection is only established during the access otherwise it is terminated. The mode in which programs in the PLC can be changed can be selected by positioning the cursor in the field after “Mode” and pressing ***F3.*** A box with possible modification modes is displayed and you can select one of the following: None: You cannot change a program on the PLC. **Stop**: You can only change a program on the PLC when the PLC is in the stop mode. **Cycl**: You can change a program on the PLC during the processing cycle.
Representation	You can select between one of the three methods of representation, LAD, CSF, STL.
PLC type	If there is a connection to the PLC, the type of PLC is specified here.

Input field	Explanation
Interface	If you press ***F3*** a number of interfaces is displayed from which you can select one. The interface AS 511 is the default. With the interfaces AS 511 and AS 511 (special) to AS 511 (special n) no bus paths are necessary. You can then select the mode. If you select a different interface, a bus path must be edited before you can select the mode.
Checksum	
Yes	When you transfer blocks to the PLC, the checksum is generated, appended to the block and transferred to the PLC. When you read from the PLC, the checksum is used to check the transfer.
No	The checksum is not generated.
Path name	Name under which an edited path (→*Bus path*) is stored. If you specify this path name and a path file, the system attempts to establish or terminate the connection stored under this path when you change modes. Successful establishment of a connection is indicated by the message PATH ACTIVE. If no connection can be established, this is indicated by the message “PLC timeout”.
Path file	Name of the file in which the path names are stored. This file is stored in the directory in which STEP 5 is installed (system directory).
Path option	
No	Files assigned to a bus path are not entered.
Confirmation	If files are assigned to a bus path, and if the path is set, the files are only entered globally in the presettings after user confirmation.
Always	If files are assigned to a bus path, and if the path is set, the files are entered globally in the presettings without user confirmation.

Input field	Explanation
Symbols	What you select here decides whether you can program with symbolic operands in all three methods of representation.
Yes	Symbolic operands with operand **comments** can be input and output in LAD, CSF, STL. You can select between a symbolic or absolute notation for the screen display and printer output.
No	The input and output of symbolic operands including comments is **not possible**.
Symbol length	In your first editing session, you can select this setting between 8 and 24 characters. After making this setting, you can increase it at any time. You can only reduce the length to that of the longest actual symbol. Before doing this, delete the *Z?.INI files.
Display	
Sym	Operands are displayed **symbolically**. If symbols are longer than 8 characters, they are truncated (only in LAD, CSF). The first 8 characters of the symbolic operand names should therefore be unique.
Abs	Operands are displayed in absolute form.
Comment length	Symbol comment. The first time you create the settings, you can select this setting with a maximum of 40 characters. You can increase the comment length at any time. You can only reduce the length to that of the longest actual comment. Before doing this, delete *Z?.INI.
Comments	
Yes	All comments are displayed. **Operand comments** are not affected by this setting.
No	Line (DB), segment and block (DBDO) comments and segment titles are not displayed.

Input field	Explanation
Documentation	
On printer	Output is printed out on the connected printer. The printer file for the particular printer must be set.
To file	Output to a selectable file (*LS.INI). You can also store a hardcopy of the screen in this ASCII file (using the **PRINT** key).
Name	Name of the *LS.INI.
Printer file	Here, you store the printer parameters selected in the “documentation” menu (→ *Printer parameters*). You select the printer type from a list and implicitly also the printer file of the type DR.INI which contains the parameters for the connected printer.
Character set	Only valid for enhanced output. The following can be selected: ASCII Documentation is only printed with the characters of the ASCII character set e.g. !– – –] [– – – – – – – – ( )– – – – ! CHAR.GRAPHICS Documentation is printed with the IBM character set e.g. |] [– ( ) |
Footer	You can decide wether or not to have a footer (80 or 132 characters wide) printed out at the bottom of the page. The “footer file” must be set if you require the latter. This file is processed in the “Documentation” menu (→ *Edit footer*). If you use the enhanced output , the 132 character long footer **must** be selected.
Diagnosis	
Yes	Default when the software for the CP 522 is not loaded. You can edit or output the diagnostic setpoint data if the CP 522 software is loaded.
No	The diagnostic setpoint data mode is disabled.

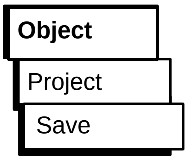
Load



With this function, you load the settings you selected under → *Project, Settings* and saved in a *PJ.INI file. All the currently valid settings are overwritten when you use the load function. As soon as you load new settings, only those in the current PJ.INI file are valid. You can, however, change these as required. The preset parameters (e.g. file names) are automatically entered in the job and selection boxes in which they are required.

The “file selection box” is displayed. You can move through this display and select a *PJ.INI file (→ Graphical user interface). After **OK**, all the settings are loaded from the *PJ.INI file.

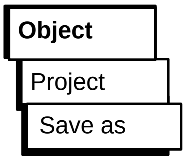
Save



With this function, you save the current settings you have made under → *Project, Settings* The settings are saved in the currently selected *PJ.INI file.

A message box is displayed in which you decide whether or not to save the settings.

Save as



With this function, you save the current settings you made under → *Project, Settings*. The settings are saved in a *PJ.INI file that you select.

The “File selection box” is displayed. You can move through this display and select a *PJ.INI file or create a new one (→ *Graphical user interface*).

3.2.2 Blocks

Object
Blocks

With the functions of this submenu, you can manage blocks and documentation files belonging to the working directory.

With these functions, you can do the following:

- output a directory (DIR)
- transfer blocks and documentation files
- compare blocks
- delete blocks and documentation files
- delete the user memory on the PLC (overall reset)

Output Directory

Object
Blocks
Directory

The following directories can be output:

From the selected program file, the block list

- of all blocks entered in the block type
- of all blocks
- of all blocks of one block type
- of all documentation files

From the programmable controller, the block address list

- of all blocks entered in the block list
- of all blocks
- of all blocks of one block type depending on the PLC type

Object
Blocks
Directory
in the program file

The following information is output for each block:

- block type, LIB NO, block length, FB name (option)

Object
Blocks
Directory
in the PLC

The following information is output for each block:

- block type, block length, address, LIB NO, FB name (option)

The job box “Blocks-directory-program file: settings” is displayed. You can browse through this box and make selections (→ *Graphical user interface*).
You can output the following:

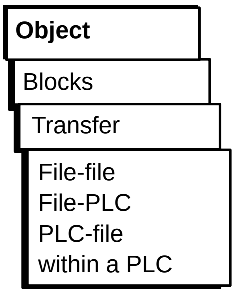
- preheaders as a group or assigned to blocks
- FBs with or without names.

Example

Once you have acknowledged the job box, the output is made on the selected output device.

The block name, the length of the block, the LIB number and corresponding FB name are listed under “block”. The character # is inserted in front of documentation blocks.

Transferring Blocks



With this function, you can transfer the following

- single blocks
- a group of blocks of one block type
- all blocks of one block type
- a group of blocks with block list
- all blocks of a program file
- one or all documentation files, known as DOCFILES
- the whole program file
- from the selected program file to a selectable drive and selectable program file (**file - file**)
- from a selectable drive with a selectable program file to the programmable controller (**file - PLC**)
- from the programmable controller to a selectable drive with a selectable program file (**PLC - file**).

Transferring Function and Data Blocks

The preheaders of these blocks contain format information and jump label information which can only be evaluated by the PG. For this reason, they are not transferred to the PLC.

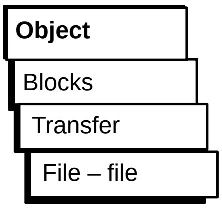
When a block that is assigned a preheader in the PG (FB/FV, FX/FVX, DB/DV, DX/DVX) is transferred, the block preheader can be deleted following a user prompt. Since the PG makes you aware of this with the message “Overwrite preheader?” no data can be accidentally lost.

By modifying a **data block (DB and DX)** during editing online in the PLC and transferring it back to the program file in the PG, the correlation between the DB (DX) and DV (DVX) may no longer exist and it is therefore often advisable to overwrite the data block preheader. The data in this data block is then displayed in the format that was previously set.

In **function blocks (FB and FX)** the names (e.g. LEVEL) of the jump labels can be lost when they are transferred back. These are then replaced by STEP 5 with substitute names, e.g. M002.

Transferring Blocks

The job box “TRANSFER block file - file” is displayed:



The following overview explains the inputs you can make in the job box that were not dealt with in the description of the job box (→ *Graphical user interface, Job box*):

Inputs	Explanation
Source	The preset program file is displayed in this field. You can edit this name or replace it with an existing file name from the file selection box using F3=Select.
Dest	The program file C: @@@@@@ ST.S5D is displayed in this field. You can edit this name or replace it with an existing file name from the file selection box using F3=Select.
Selection	
all doc files	If you mark this parameter with an “X” all DOCFILES are selected. Docfiles are indicated by the character # and can be stored. Documentation files cannot be stored on the PLC
whole file	By marking the field with an “X” you can select the whole program file (including docfiles).
Block area	If you want to transfer several successive blocks of one block type, mark the line and enter the first block identifier (e.g. **PB7**) in the field “from” and the last block identifier (e.g. **PB22**) in the field “to”.
Copy	If you want to copy a single block and store it under a different name, mark the line and enter the source block in the “Block” field (e.g. **DB6**) and the new block identifier (e.g. **DB 54**) in the “to” field. When you copy blocks you must not change the block type.
Transfer (**OK**)	The PG transfers the selected blocks. If errors occur during the transfer, you will see alternatives displayed in the selection boxes and you can select the best course of action in the situation.

Object
Blocks
Transfer
File – PLC

The job box “TRANSFER block: file - PLC” is displayed. You can browse through this box and make your selection (→ *Graphical user interface*).

When transferring to the PLC, remember that you can only transfer block types that can be selected in the job box. If you select an illegal block type, the transfer request will be denied.

For the description of specific parameters, refer to the explanations for “file - file”.

Object
Blocks
Transfer
PLC - File

The job box “TRANSFER block: PLC - file” is displayed. Here you can browse and make your selections (→ *Graphical user interface*).

Note

Not all blocks that can be displayed can be transferred; this depends on the PLC.
Only blocks up to a maximum of 4 Kw (8 Kb) can be transferred.

Comparing Blocks

Object
Blocks
Compare
File-file
File-PLC
PLC-file

With this function, you can compare the following:

- a block, a group of single blocks or all blocks of the first named program file with those of the second named program file.

The comparison operation is between the program file preset on the PG and any other program file or the user program on the PLC. It is also possible to compare the program in the PLC with a selectable program file.

Note

Data blocks you want to compare must not be larger than 2 Kwords.

Object
Blocks
Compare
File – file

The job box “Block COMPARISON: ....” is displayed. Here, you can browse through the box and make your selection (→ *Graphical user interface, Job box*).

When you compare blocks in the PLC, remember that you can only compare blocks that can be selected in the job box.

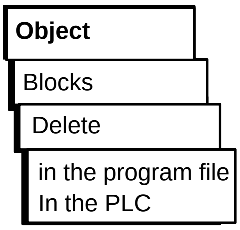
Object
Blocks
Compare
File – PLC

In this case the PLC type and CPU identifier are displayed in the job box.

The job box “Block COMPARISON: ....” is displayed. Here, you can browse through the box and make your selection (→ *Graphical user interface, Job box*).

Object
Blocks
Compare
PLC - File

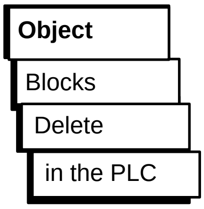
Delete



With this function you can delete the following:

- single blocks
- a group of blocks of one block type
- all blocks of one block type
- all blocks (only PLC: overall reset)
- one or more documentation files (only in the PG)
- the whole program file (only in the PG)

The job box “DELETE block(s)” is displayed. Here, you can browse through the box and make your selection (→ *Graphical user interface*).



If all the blocks in the PLC are deleted, this corresponds to the OVERALL RESET function (only possible in the STOP mode). The PLC sets defined statuses in the PLC RAM (see programming guide for the relevant PLC).

3.2.3 DOS Files

With the functions in this submenu, you can manage files without returning to the operating system level. The following functions are available:

- Output single files or groups of files from the currently selected directory on the screen.
- Copy single files or groups of files (source file name # destination file name).
- Delete single files or file groups in the currently selected directory.

Significance of the wildcards

? A question mark can stand for any character within a file name.

* An asterisk can only be the last or the only character in a file name or file extension. The operating system replaces the asterisk by one or more question marks up to the end of the file name or file extension.

Directory

Object

DOS file

Directory

This function displays the directory of

- a file
- several files.

The job box “DOS-files-directory” is displayed. Here, you can browse through the box and make your selection (→ *Graphical user interface, Job box*).

Inputs	Explanation
File name	The file name marked by the cursor in the list of field names is displayed here. This is not a field for direct input.
Search key	If you want to find a particular file or group of files, you can enter the name here. Wildcards are allowed, for example ??????.INI. Files matching the search key are displayed in the “Files” field.
Drive/dir	Here, you can select a drive and a directory on this drive. Once you have made the selection, the contents of the directory appears in the display field.

Copying

Object

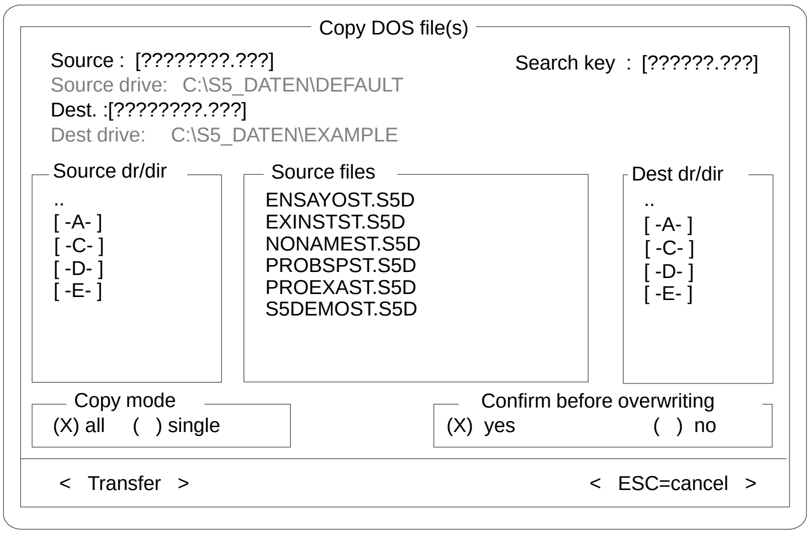
DOS file

Copy

This function copies one or more files between different drives or directories.
With the copying function, you can either

- retain the file name or
- use a different file name.

The following job box is displayed. Here, you can browse through the box and make your selection (→ *Graphical user interface*, *Job box*).
In the example below, all the files of the type *ST.S5D are copied from the directory C:\S5_DATEN\DEFAULT to the directory C:\S5_DATEN\EXAMPLE:



Copy DOS file(s)

Source : [????????.???]
Search key : [??????.???]
Source drive: C:\S5_DATEN\DEFAULT
Dest. :[????????.???]
Dest drive: C:\S5_DATEN\EXAMPLE

Source dr/dir
..
[-A-]
[-C-]
[-D-]
[-E-]

Source files
ENSAYOST.S5D
EXINSTST.S5D
NONAMEST.S5D
PROBSPST.S5D
PROEXAST.S5D
S5DEMOST.S5D

Dest dr/dir
..
[-A-]
[-C-]
[-D-]
[-E-]

Copy mode
(X) all ( ) single

Confirm before overwriting
(X) yes ( ) no

< Transfer > < ESC=cancel >

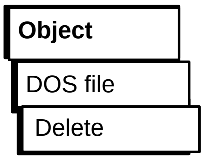
Figure 3-13 Copying DOS Files

Input field	Explanation
Source	Name of the file you want to transfer.
Search key	If you want to find a particular file or group of files, you can enter the name here. Wildcards are allowed, for example ??????.INI. Files matching the search key are displayed in the “Files” field.
Source drive	Drive and directory as set in “Source dr/dir” from which the file is transferred.
Dest	Name of the destination file.
Dest drive	Drive and directory as set in “Dest dr/dir” to which the file is transferred.
Source dr/dir	Here, you select a source drive and directory. This is displayed in the “Source drive” field.
Source files	Displays the files that exist on the source drive. You can only select with the cursor/ mouse click. All the files are then displayed if question marks (or *.*) are entered in the “Search key” field.
Dest dr/dr	Here, you select a destination drive and directory. This is displayed in the “Dest drive” field.
Copy mode	
all	all files in the “Source files” field are copied.
single	one file entered in the “Source” field or “Source files” field and selected with the cursor is copied..
Transfer	The function is executed.

Procedure

1. In the field “Source dr/dir”, select the drive and directory from which you want to transfer (copy) one or more files.
2. In the “Dest dr/dir” field, select the drive and directory to which you want to transfer file(s).
3. You can either transfer single files or all the files listed in the “Source files” field.
   <u>Single files:</u> Either type the name of the file in the “Source” field (no wildcards permitted) or select the file in the “Source files” field by clicking with the mouse and click “single” in the “Copy mode” field.
   <u>Several files:</u> Click “all” in the “Copy mode” field and specify whether you want to transfer all the files of the source directory or only a selection in the “Search key” field. If you specify ??????.??? or *.*, all the files are displayed and transferred. If, for example, you only want to transfer STEP 5 program files, type in *ST.S5D as the search key.
4. If you want to save the destination files under a different name, type in the new name or a group name.
   If, for example you specified *.DOC as the search key for the text files to be transferred, you could for example specify file type *.TXT in the “destination” field.
5. Click “transfer” to start the copying.

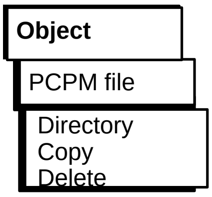
Deleting Files



This function deletes files according to the selected delete mode (single or all) in a selected directory.

The job box “Delete DOS file(s)” is displayed. You can move through this box and make your selection (→ *Graphical user interface, Job box*)

3.2.4 PCP/M File



The following functions are available:

- Conversion of PCP/M files to S5-DOS ST/MT files. Under the latter operating system they can then be run and processed.
- Conversion of STEP 5 files created with S5-DOS/ST or S5-DOS/MT to PCP/M files. You can then run these converted files and process them under the S5-DOS operating system.

Here, you have functions available to process PCP/M media. PCP/M media are disks formatted under S5-DOS (PCP/M).

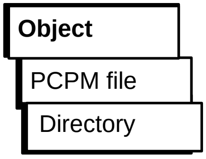
The following functions are available:

→ Output *directory* of PCP/M files from selectable user areas

→ *Copy* (convert) the following from or to PCP/M media:
– PCP/M files to DOS files
– DOS files to PCP/M files

→ *Delete* PCP/M files

Directory



Outputting a file list of a selectable USER area from a PCP/M disk.

The job box “PCP/M file directory” is displayed. Here you can browse and make your selections (→ *Graphical user interface, Job box*). Depending on your input, a directory known from PCP/M is displayed in a window:

File name	STEP5 files (e.g. *F1.INI)
Bytes	Number of bytes in the file
Recs	Number of records
Attrib.	File access mode

Input field	Explanation
File name	The file name marked by the cursor in the directory is displayed here. This is only an output field and cannot be changed.
Search key	If you want to find a particular file or group of files, you can enter the name here. Wildcards are allowed, for example ??????.INI. Files matching the search key are displayed in the “Files” field.
Drive	Drive containing the files. This field is only for information and no inputs can be made in it.
User area	USER area in which the source is located. This field is only for information and no inputs can be made in it.
Files	Display of the files in the USER area on the selected drive. No input can be made here.
Drives	All the existing PCP/M drives are displayed. You can select one from this list.
User area	List of all USER areas. You can select one of the user areas in this list.

Copying

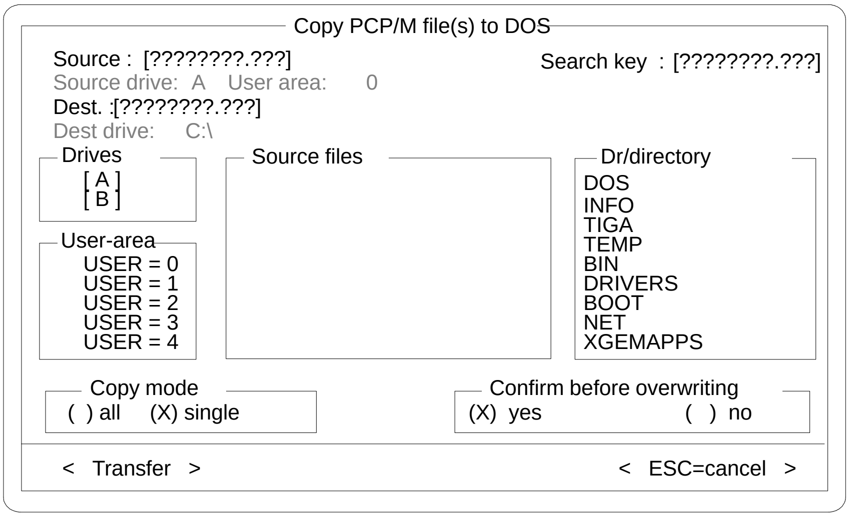
Object
PCP/M file
Copy

With this function, you can convert the following:

- PCP/M files to S5-DOS/ST/MT files
- S5-DOS/ST/MT files to PCP/M files

Object
PCP/M file
Copy
PCP/M file –>DOS file

The following job box is displayed. You can move through this box and make your selection (→ *Graphical user interface, Job box).*



Copy PCP/M file(s) to DOS

Source : [????????.???]
Search key : [????????.???]
Source drive: A User area: 0
Dest. :[????????.???]
Dest drive: C:\

Drives
[A]
[B]

User-area
USER = 0
USER = 1
USER = 2
USER = 3
USER = 4

Source files

Dr/directory
DOS
INFO
TIGA
TEMP
BIN
DRIVERS
BOOT
NET
XGEMAPPS

Copy mode
( ) all (X) single

Confirm before overwriting
(X) yes ( ) no

Figure 3-14 Copying PCP/M File(s) to DOS

Explanation of the job box

Input field	Explanation
Source	Name of the file to be transferred.
Search key	If you want to find a particular file or group of files, you can enter the name here. Wildcards are allowed, for example ??????.INI. Files matching the search key are displayed in the “Files” field.
Source drive	Drive from which the file is transferred.
User area	User area of the source. This field is only for information, no input possible.
Dest	Name of the destination file.
Dest drive	Drive to which the file is transferred. This field is only for information, no input possible.
Drives	Here, you select a source drive. This is displayed in the “Source drive” field.
User area	Here, you select a USER area. This is displayed in the “User area” field.
Source files	Display of the files existing on the source drive You can select files with the cursor or mouse click. All files are only displayed when question marks (or *.*) are entered in the “Search key” field.
Dest dr/dir	You select a destination drive or directory.
Copy mode	Copies
all	all files displayed in the “Source files” field.
single	one file entered in the “Source” field or selected in the “Source files” field.
Transfer	The function is executed.

Copying

The following job box is displayed. You can move through this box and make your selection (→ *Graphical user interface, Job box).*

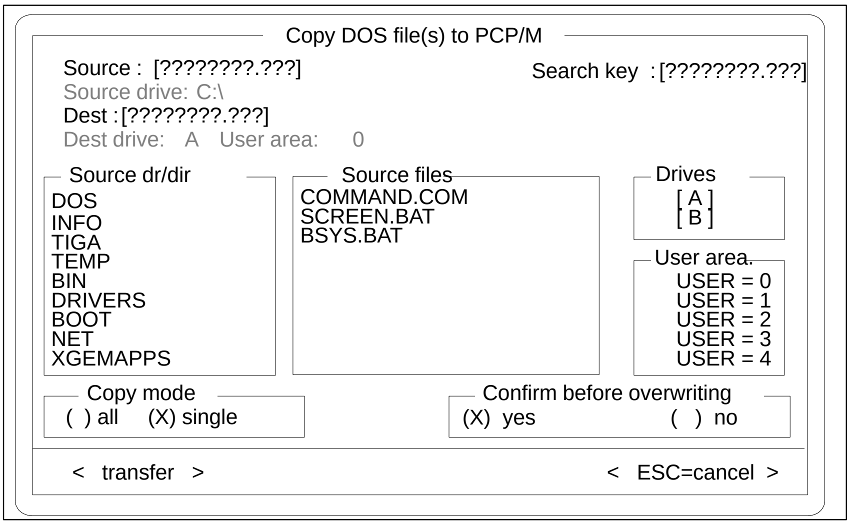
The explanation of the input fields on page 3-46 for copying PCP/M files to DOS files also apply to this function. Only the position of the fields in the selection box is different. On the left the source (dr/dir), on the right the destination (drive, user area).

Object

PCPM file

Copy

DOS file –>PCPM file



Copy DOS file(s) to PCP/M

Source : [????????.???]

Search key : [????????.???]

Source drive: C:\

Dest : [????????.???]

Dest drive: A User area: 0

Figure 3-15 Copying DOS File(s) to PCP/M

Deleting

Object
PCPM file
Delete

PCP/M files are deleted on a PCP/M medium. You can delete a single file or all files in a USER area.

The selection box “Delete PCP/M file(s)” is displayed. You can move through this box and make your selection (→ *Graphical user interface, Job box)*. The significance of the input fields is the same as for the PCP/M file directory.

Note

All files in a USER area are only displayed when question marks are entered in the “Search key” field.

3.2.5 End

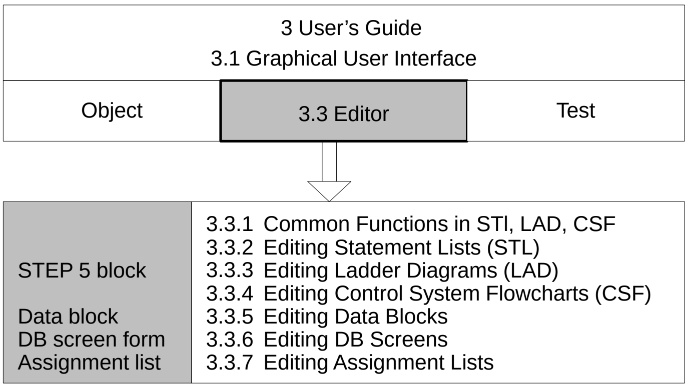
End

Object
End

With this function, you terminate STEP 5. You are prompted to confirm that you want to exit the program so that you cannot terminate it accidentally.

– Answer with “End” if you really want to exit STEP 5.
– Answer with “Cancel” if you want to return to the user interface.

3.3 Editor



3 User's Guide
3.1 Graphical User Interface

Object | 3.3 Editor | Test

STEP 5 block
Data block
DB screen form
Assignment list

3.3.1 Common Functions in STl, LAD, CSF
3.3.2 Editing Statement Lists (STL)
3.3.3 Editing Ladder Diagrams (LAD)
3.3.4 Editing Control System Flowcharts (CSF)
3.3.5 Editing Data Blocks
3.3.6 Editing DB Screens
3.3.7 Editing Assignment Lists

Figure 3-16 STEP 5 Editors

With the STEP 5 programming language you describe automation tasks in the form of user programs which can be run on SIMATIC S5 programmable controllers. With the STEP 5 editors, you can edit a complete user program as follows:

- blocks in the methods of representation **STL, LAD, CSF** with segment titles, statement and plant comments,
- data blocks with block titles, data word and block comments,
- DB screen forms for the S5-135U and S5-155U,
- symbolic operands with operand comments in the assignment list. This is translated to a symbols file following the editing session.

Internally, STEP 5 converts the blocks OB, PB, SB, FB, FX, DB, DX into the MC 5 machine code and stores them on the hard disk, on diskette or in the PLC.

STEP 5 blocks and editors

	Object to be edited	
Editing function	**..in the program file**	**..in the PLC**
STEP 5 block, STEP 5 block with search	OB, PB, SB, FB, FX in the block: Segment title, segment comment, symbolic operands and operand comments	OB, PB, SB, FB, FX
STEP 5 block	OC, PC, SC, FC, FCX	——
STEP 5 block	OBDO, PBDO, SBDO, FBDO, FXDO, DBDO, DXDO, plant comm.	——
Data block, DB with search	DB, DX in the block: block title, comment, DB, DX	DB, DX
Data block	DC, DCX	——
Data block	DBDO, DXDO, plant comment	——
DB screen form	DB1, DX0	DB1, DX0
Assignment list	symbolic operands and operand comments in the Z0.SEQ (translated to *Z0.INI when stored)	——

3.3.1 Common Functions in STL, LAD, CSF

This section lists all the functions you can use when editing in the three methods of representation. These include the following:

- Selecting an editor
- Selecting an editor with the search function
- Assignment of the function keys in the output mode
- Inputting a library number
- Switching over the method of representation
- Editing comments
- Displaying operand comments
- Working with segments
- Displaying cross references, changing blocks
- Searching for operands
- Editing symbolic operands in a block

Selecting the Editor

Editor

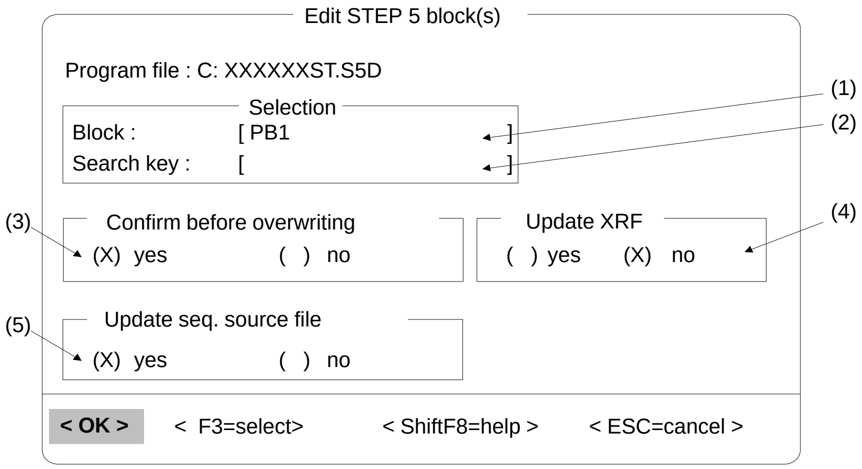
STEP 5 block

in the program file
in the PLC

Editing STEP 5 blocks in the methods of representation LAD, CSF or STL and comment blocks, documentation blocks and plant comments. The method of representation depends on the “setting”
(→ *Project*) but this can be changed in the editing mode using a function keys.

(1) To edit a block, you enter it in absolute or symbolic form. A search key (2) does not need to be entered.

If you want to search for a term in one or more blocks, enter the block or blocks (maximum 6) in the block list in absolute form or a block in symbolic form. The search key, e.g. I 1.1 must be entered in (2). If the block entered here does not exist, the first empty segment of this block is displayed in the “editing” mode once you have entered the parameters and options.



Edit STEP 5 block(s)

Program file : C: XXXXXXST.S5D

Selection

Block : [PB1]

Search key : [ ]

Confirm before overwriting

(X) yes ( ) no

Update XRF

( ) yes (X) no

Update seq. source file

(X) yes ( ) no

Figure 3-17 Editing STEP 5 Block(s)

If you press ***SHIFT F8*** = *Help*, STEP 5 displays a list of possible inputs. If you want to edit an existing block, you can call the block with ***F3*** = *Select* in the → *Block selection box*.

Selection/Search Key

(2) Here, you enter a search key in absolute or symbolic form. You can display the permitted search keys by pressing ***SHIFT F8*** = *Help*. You exit the field with the ***Return*** key or you can select a different field with the mouse. The key is searched for in all the specified blocks. When the key is found, the segment is displayed in the output mode.

You can continue the search for the key as follows:

- in the following segments with ***F3*** = *Search*,
- in the next blocks with the ***Insert*** key confirmed by the ***Return*** key.

Confirm Before Overwriting

(3) yes When you store changes, you are first prompted to confirm the changes within the individual blocks:

program block,
comment block,
documentation block,
documentation file.

no Modified blocks are overwritten as soon as you enter the changes. In program blocks OB, PB, SB, FB/FX you are always prompted to confirm the changes.

Update XRF

(4) yes The cross reference list (file *XR.INI) is updated or set up if it does not yet exist.

no The cross reference list is not updated. You can, however, update or set up the cross reference list later using the function → *Generate XRF.*

Update Seq. Source File

(5) yes (only displayed if you selected symbols: yes)
If you want to edit symbolic operands, i.e. change the symbols file *Z0.INI, the sequential source file *Z0.SEQ is updated when you store your input.

no You can update or set up the sequential file later using the function → *INI > SEQ*.

Selecting an Editor with Search

Editor

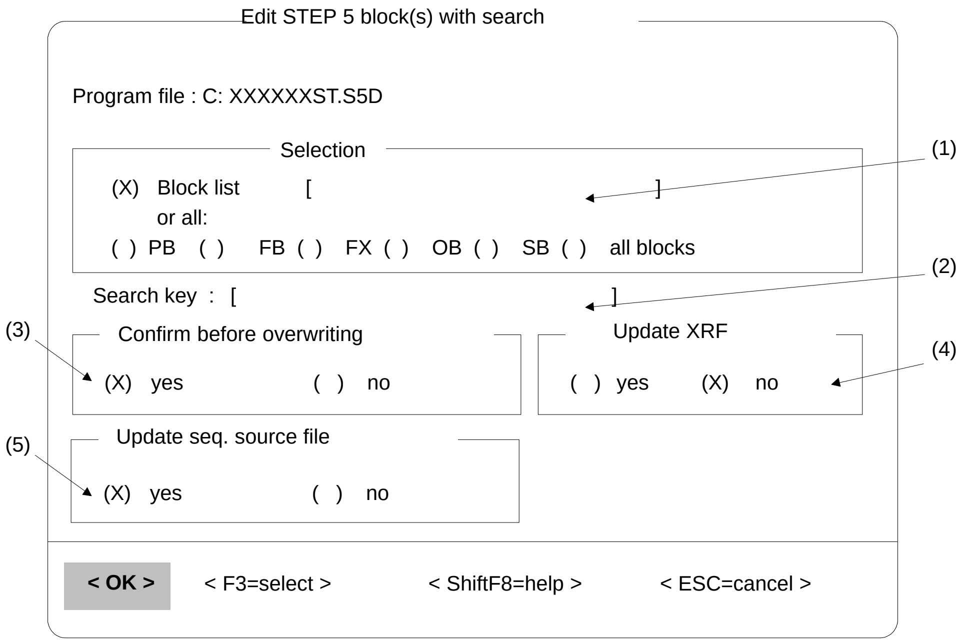
STEP 5 block >

Search
in the program file
in the PLC...

You can specify comment blocks, documentation blocks and documentation files but they will be rejected since it is not possible to search in these blocks.

Editing STEP 5 blocks in the methods of representation LAD, CSF, STL with search. The key is searched for automatically in all the specified blocks.

The method of representation depends on the setting (→ *Project*), but can also be changed while editing in the output mode (→ *Switching over the method of representation*).



Edit STEP 5 block(s) with search

Program file : C: XXXXXXST.S5D

Selection

(X) Block list [ ]
or all:
( ) PB ( ) FB ( ) FX ( ) OB ( ) SB ( ) all blocks

Search key : [ ]

Confirm before overwriting
(X) yes ( ) no

Update XRF
( ) yes (X) no

Update seq. source file
(X) yes ( ) no

Figure 3-18 Editing STEP 5 Blocks with Search

Note

If you select an editor with search and modify the block, the modified block must first be saved before you can continue searching.

1. Press ***F7*** = ***OK*** in the EDIT mode and confirm the message “Enter modified segment?” with “**Yes**”.
2. The editor changes to the output mode. Now press ***F7*** = ***OK*** and confirm the message “Enter modified block?” with “**Yes**”. If you confirm the message “Continue” with “**Yes**” the search is continued, if you reply “**No**” you return to the main menu.

Selection/Block List

(1) You enter one or more blocks (max. 6) in the block list in absolute form separated by a comma or one block in symbolic form. The search key, e.g. I 1.1 must be entered in (2).

If the first block entered does not exist, the first empty segment of this block is displayed in the “edit” mode after you enter the parameters and options. Once you exit this block, the key is searched for in the other blocks you have specified. If there is another non-existent block in the list, this is skipped during the search.

If you press ***SHIFT F8*** = *Help* STEP 5 displays a list of possible inputs. You can select existing blocks with ***F3*** = *Select* in the *Block selection box.*

( ) PB ... ( ) SB ( ) all blocks
Here, you can specify a block type or all blocks in which case you do not need to make an entry in the block list.

Selection/Search Key

(2) Here, you enter a search key in absolute or symbolic form. If you press ***SHIFT F8*** = *Help* the permitted search keys are displayed. You exit the field with the ***Return*** key or select a different field with the mouse. The key is searched for in all the specified blocks. The segment containing the search key is displayed in the output mode.

You can continue the search for the key as follows:

- in the following segments with ***F3*** = *Search*,
- in the next blocks with the ***Insert*** key confirmed by the ***Return*** key.

Confirm before Overwriting

(3) → Job box: “Edit STEP 5 blocks”

Update XRF

(4) → Job box: “Edit STEP 5 blocks”

Update Seq Source File

(5) → Job box: “Edit STEP 5 blocks”

Assignment of the function keys in the output mode

This section contains the functions you can use when editing regardless of the method of representation. This includes the functions in the output mode and those for editing comments. The following description of the keys provides you with an overview of the tools and functions available to support editing.

Function keys

F	Addresses	F	Lib. no.	F	Symb. SYM	F	No comm.
1	Disp. Symb.	2	Reference	3	Search	4	Diagnosis

F	–> LAD	F	**Seg comm.**	F	Save	F	Help
5	Seg fct	6	Edit	7	Enter	8	Cancel

F1
= Disp symb
Edit symbolic operands directly in the block.

F2
= Reference
Display references (cross references), change block.

F3
= Search
Search for single operands.

F4
= Diagnosis
Setpoint data output: this function is required for the CP 522 software package for handling setpoint data.

F5
= Seg fct
Copy, mark, insert, append and delete segments.

F6
= Edit
Change to the edit mode, also possible with the ***CORR*** key.

F7
= Enter
Store the block if this has been changed, or return to the main menu.

F8
= Cancel
Return to the main menu.

SHIFT F1
= Addresses
Display relative operation addresses in bytes or words; only in STL (→ *Editing Statement Lists Displaying addresses*)

SHIFT F2
= Lib no
Input library number

SHIFT F3 *= Symb. SYM/ABS/OFF*	Switch symbols on and off.
SHIFT F4 *= No com*	Switch line and symbol comments on and off.
SHIFT F5 *= → LAD*	Switch over to the indicated method of representation, LAD, CSF or STL.
SHIFT F6 *= Seg com*	Edit the segment title or segment comments.
SHIFT F7 *= Save*	Save block without confirmation. You do not exit the editor.
SHIFT F8 *= Help*	Explains the function keys.

Inputting the Library Number

The library number is a 5 digit number (0 to 99999) to identify blocks.

Ready to Start?

The block in which you want to enter the library number is open. STEP 5 is in the output mode.

How to Input the Library Number

1. Press ***SHIFT F2*** *= Lib no*
   The cursor is located in the displayed LIB field.
2. Type in the required LIB no or modify the existing LIB no.
3. To exit the LIB field: press the ***Return*** key.

If you enter 5 numbers the cursor automatically leaves the library number field. If you do not enter a number or a number with less than 5 digits, exit the field with ***ESC*** or the ***Return*** key.

Method of Representation

With this function you can switch over the method of representation without having to call up the settings (→ *Project*).

Ready to Start?

STEP 5 is in the output mode. The displayed segment must be capable of translation into the required method of representation.

How to Change the Representation

*Press **SHIFT F5** = → LAD or click on it with the mouse.*

The segment now appears on the screen as a Ladder Diagram. If the segment cannot be represented in LAD or CSF, STEP 5 displays the message “LAD/CSF segment not translatable”.

The function key display is now “→ CSF”. If you press ***SHIFT F5*** again, the segment is displayed as a Control System Flowchart and the softkey display is “→ STL”.

Editing Comments

You can add the following comments to the STEP 5 blocks OB, PB, SB, FB and FX:

- Plant comments
- Statement comments (→ *Editing Statement Lists*)
- Segment comments
- Segment titles
- Operand comments (→ *Editing Assignment Lists*)

Comments for data blocks DB and DX can be found in the section → *Editing data blocks*.

Type of comment	Where can you edit it?	Where is it stored?
Plant comment	Documentation file	# Documentation file
Statement comment	STL : OB, PB, SB, FB, FX Documentation block: OC, PC, SC, FC, FCX	OC, PC, SC, FC, FCX
Segment comment	STL, LAD, CSF : OB, PB, SB, FB, FX Documentation file: #OBDO.nnn, #PBDO.nnn, #SBDO.nnn, #FBDO.nnn, #FXDO.nnn	#OBDO.nnn, #PBDO.nnn, #SBDO.nnn, #FBDO.nnn, #FXDO.nnn
Segment title	STL, LAD, CSF : OB, PB, SB, FB, FX Documentation block: OC, PC, SC, FC, FCX	OC, PC, SC, FC, FCX
Operand comment	STL, LAD, CSF : OB, PB, SB, FB, FX Assignment list	*Z0.INI *Z0.SEQ

Plant Comment

Editor
STEP 5 block
in the program file

A plant comment is a text file (documentation file) and in contrast to the segment comment is not oriented to one block. With the S5-DOS data management, the number of characters of all the plant comments in a program file must not exceed 16 K characters. The maximum number of documentation files in a program file is 255.

A plant comment is stored on diskette or on hard disk and is not transferred to the PLC or to the EPROM/EEPROM.
When editing the plant comment, you can call up the command mode and editing aids for text processing.

Name

The name begins with the # character, following this, the name can have a maximum 8 further characters, e.g. #EXAMPLES. When you type in a plant comment, you must make sure that the second character of the name is not a colon, otherwise STEP 5 will **not** store the file, since it attempts to identify the letter before the colon as a drive.

Working with the Editor

1. Select STEP 5 block or data block in the editor menu.
2. Type in the name of the documentation block preceded by the character *"#"* and enter your selection (→ *Job box*).

Type in your texts using the alphanumeric keyboard. The text editor includes the following functions:

F	F	F	F	F
1 Insert	2 Delete	3 Command	4	5 Insert l.

F1 *= Insert*	Switchover between the insert and overwrite modes. The selectable mode is displayed.
F2 *= Delete*	Delete a string of characters in the text.
F3 *= Command*	Commands for fast text processing.
F5 *= Insert l*	Insert a line at the cursor position.
F6 *= Delete l*	Delete the line at the cursor position.

Inserting Text

You can insert ASCII characters within a text.

1. Press ***F1*** = *Insert.*
2. Type in the required string.
3. Switch over to the overwrite mode by pressing ***F1*** = *Overwrite.*

The insertion of the text is completed.

Deleting Text

Within a text, you can delete character strings and sections of text of any length.

1. Position the cursor on the first character you want to delete.
2. Press ***F2*** = *Delete.*
   STEP 5 displays the start marker @ at the cursor position.
3. Position the cursor after the last character you want to delete.
4. Press ***F2*** = *Delete* again.
   The text between markers is deleted. The remaining text is automatically repositioned.

Commands

The text editor has 8 commands for fast text processing.

Note

Write commands in upper case letters. The printer control character **$EJECT** triggers a form feed in a segment, block or plant comment.
$EJECT must also be in upper case letters, otherwise STEP 5 does not recognize the command. If you only type the dollar sign, the segment comment is not printed out beyond this point.

How to Use the Commands

You call the command mode by pressing ***F3*** = *Command.*
The keystrokes for all commands are the same:

1. Position the cursor in the text.
2. Press ***F3*** = *Command.*
3. *T*ype in one of the 8 possible commands.
4. Press the ***Return*** key and the ***Insert*** key.

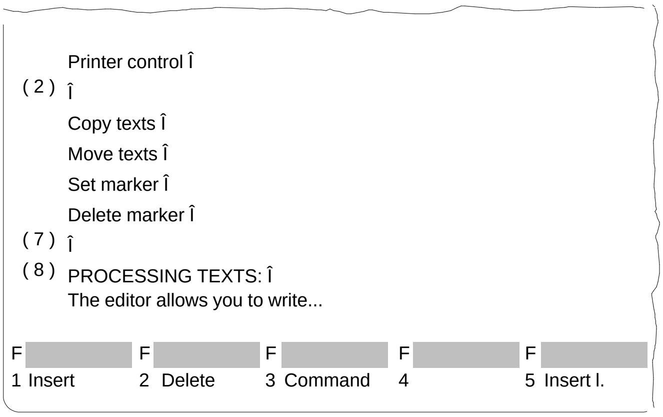
The PG executes the command.

The 8 Commands

1. **JTT** (jump to the top).
   From any position, the cursor jumps to the start of the comment.
2. **JTE** (jump to the end).
   From any position, the cursor jumps to the end of the comment.
3. **ST1, ST2, ST3, ST4** (set tag 1 etc.).
   You can set a maximum of 4 tags within the text.
4. **JT1, JT2, JT3, JT4** (jump to tag 1 etc.).
   From any position in the text, the cursor jumps to the specified tag.
5. **F/xyzrst/** (find).
   The cursor jumps to the selected text **xyzrst**, otherwise STEP 5 displays the message “not found”.
6. **CTm, Tn** (copy , m and n represent the numbers 1, 2, 3 or 4).
   You copy the text from tag Tm (inclusive) to tag Tn, the cursor must not be located between the two tags. Otherwise, STEP 5 displays the error message “Illegal between tags”.
   When you copy text, the tags are copied along with the text.
7. **MTm, Tn** (move, m and n represent the numbers 1, 2, 3 or 4).
   The text from tag Tm (inclusive) to tag Tn is moved. The cursor must not be located between the two tags. Otherwise STEP 5 displays the error message “Illegal between tags”.
   When you move text, the tags are moved along with the text.
8. **DT1, DT2, DT3, DT4** (delete).
   You can delete tags in any order.

Example

Copying a text:
You want to copy the empty line (7) and the title in line (8) into line (2).



```
        Printer control î
( 2 )   î
        Copy texts î
        Move texts î
        Set marker î
        Delete marker î
( 7 )   î
( 8 )   PROCESSING TEXTS: î
        The editor allows you to write...
```

First, you must select the text you want to copy by setting the start and end tags.

Defining the Start

1. Position the cursor on the arrow in line (7) and press ***F3*** *= Command.*

The cursor jumps to the top left corner of the screen.

2. Type in the characters *“ST1”* and press the ***Return*** and ***Insert*** keys.

The cursor returns to the text.

Defining the End

3. Position the cursor after the last character (here arrow) in line (8) and press ***F3***.

The cursor returns to the top left-hand corner of the screen.

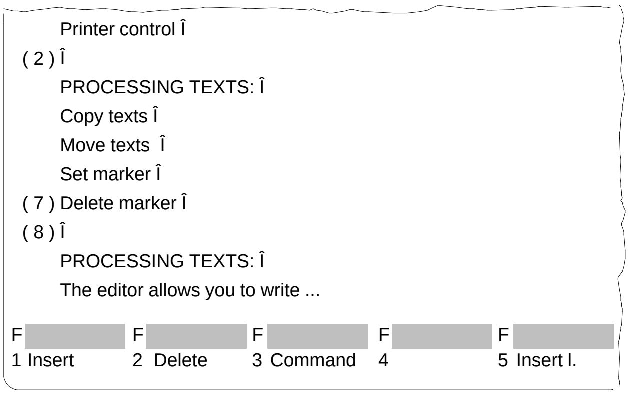
4. Type in the characters “ST2” and press the ***Return*** and ***Insert*** keys.

The cursor returns to the text.

Copying a Block of Text

5. Position the cursor on the arrow in line (2) and press ***F3***.
6. Type in the characters “CT1,T2” and press the ***Return*** and ***Insert*** keys.

The selected section of text including the empty line is inserted in line (2) as shown in the following figure. The tags are at the beginning and end of the copied text.



```
         Printer control Î
 ( 2 ) Î
         PROCESSING TEXTS: Î
         Copy texts Î
         Move texts  Î
         Set marker Î
 ( 7 ) Delete marker Î
 ( 8 ) Î
         PROCESSING TEXTS: Î
         The editor allows you to write ...

F              F              F              F              F
1 Insert       2  Delete      3 Command      4              5  Insert l.
```

Moving a text
With this function, a marked block of text is moved and the gap left by the text is closed automatically. The text marked for copying is moved to the current cursor position using the command “MT1,T2” followed by the ***Return*** and ***Insert*** keys.

Segment Comment

Editor
STEP 5 block
in the program file

Segment comments are texts with which you can write extra information about programs in segments or blocks. With the S5-DOS data management, the number of characters in all the segment comments in a program file must not exceed 16 K characters per block. The maximum number of possible documentation blocks in a program file is 255.
It is best to edit segment comments directly in the blocks and not in the documentation blocks. If you want to edit comments in documentation blocks, follow the procedure outlined in the section
→ *comments, plant comment*.

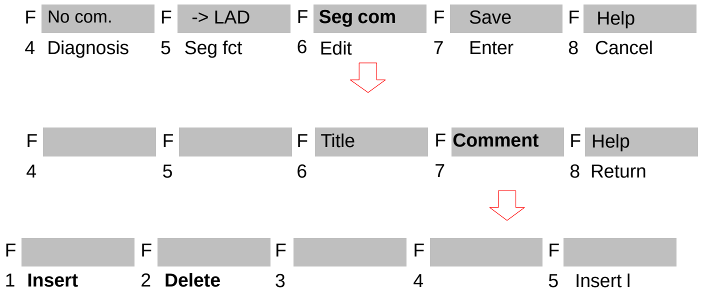
- The block and documentation file are stored in the program file.
- Documentation files cannot be transferred to the PLC or to an EPROM/EEPROM submodule.
- The block number and the number of the documentation file are the same, e.g. #PBDO.013 belongs to PB 13.
- Each block type has a corresponding documentation file in each case preceded by the character “#”:
  OBn → #OBDO.nnn
  PBn → #PBDO.nnn
  SBn → #SBDO.nnn
  FBn → #FBDO.nnn
  FXn → #FXDO.nnn

Note

You trigger a form feed with the printer control character **$EJECT.** This string must be written in upper case letters, otherwise STEP 5 does not recognize the command. If you only write the dollar sign, the text following the dollar sign will not be printed out.

Ready to Start?

You have selected “comments: yes” in the settings (→ *Project*) or by pressing **SHIFT F4** in the editor. The segment for which you want to write a segment comment is open. STEP 5 is in the output or edit mode.



Working with the Editor

1. *Press* ***SHIFT F6*** *= Seg com and* ***SHIFT F7*** *= Comment or press the* ***COM*** *key twice.*

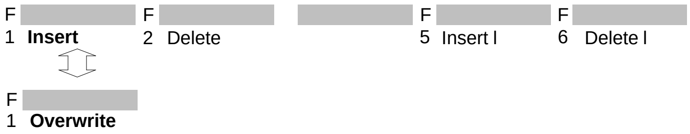
STEP 5 opens the empty editing field for the segment comment or displays text you have already input. To allow the comment to be assigned to the segment, STEP 5 generates a 7 character string “$1 @” with the number of the segment. Do not delete this number, otherwise the connection between the segment and comment is lost.

2. Edit the text using the alphanumeric keyboard.
3. Complete each line with the ***Return*** key.

The end of the line is marked by a vertical arrow.

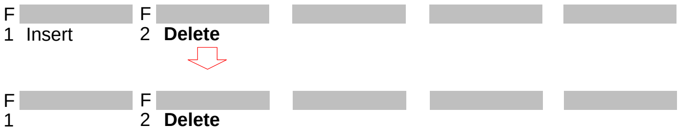
If your text takes more than one line, a line break is set at the end of the line automatically.

Inserting Characters



1. Position the cursor on the position in the text from which you want to insert characters.
2. Press ***F1*** = *Insert.*
3. Insert text.
4. Press ***F8*** = *End* to complete inserting text.

Deleting Characters



1. Position the cursor on the first character to be deleted.
2. Press ***F2*** = *Delete.*
3. Position the cursor after the last character to be deleted.
4. Press ***F2*** = *Delete.*

Inserting a Line

F 1 Insert
F 2 Delete
F 5 Insert l
F 6 Delete l

1. Position the cursor in the line before which you want to insert an empty line.
2. Press ***F5*** or click the "***Insert l***" field.

Deleting a Line

F 1 Insert
F 2 Delete
F 5 Insert l
F 6 Delete l

1. Position the cursor in the line you want to delete.
2. Press ***F6*** or click the "***Delete l***" field.

Completing the Segment Comment

Press ***F8*** = *Return.*

STEP 5 displays the corresponding segment on the screen. The text entered up to now is retained. When you store the block, STEP 5 also stores the segment comment.

Storing the Segment Comment

*Press the **Insert** key.*

Segment Title

Editor

STEP 5 block

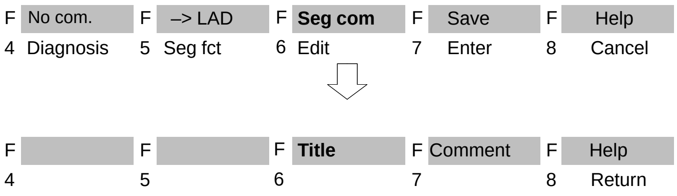
in the program file

With the segment title, you can identify a segment. A segment title has a maximum of 32 characters. You can enter it directly in the block or separately in the corresponding comment block. The first method is advisable, since the assignments are automatically updated if you make changes and store the segment. STEP 5 stores the segment title in the comment block.

– The comment block is stored in the preset program file.
– Comment blocks cannot be transferred to the PLC or to an EPROM/EEPROM submodule.
– The block number and the number of the comment block are the same, e.g. PC 13 belongs to PB 13.
– STEP 5 assigns the name of the comment block immediately as follows:
  OBn → OCn
  PBn → PCn
  SBn → SCn
  FBn → FCn
  FXn → FCXn

Ready to Start?

You have selected “comments: yes” in the “Settings” (→ *Project*).
If this is not the case, you can switch over by pressing ***SHIFT F4*** = *Line comm.* The segment in which you want to enter a title is open. STEP 5 is in the output or edit mode.



Inputting the Segment Title

1 Press ***SHIFT F6*** = *Seg com* and ***SHIFT F6*** = *Title* or press ***COM*** and ***SHIFT F6*** = *Title.*

The cursor jumps to the input field of the segment title.

2. Type in text or correct an existing text
3. Press the ***Return*** key.

The title is buffered, but is only stored in the comment block in the program file when the block is stored.

Operand Comments

When a segment is open, you can display the operand comments for symbolic operands at any time.

Ready to Start?

The symbols file is entered in the *settings* and “*symbols: yes*” and “*display: sym*” are selected. If this is not the case, you can switch over by pressing ***SHIFT F3*** = ***Symb SYM***.

Display in LAD/CSF

Position the cursor on a symbolic operand in the segment. The symbolic operand with the operand comment is displayed in the third screen line.

Display in STL

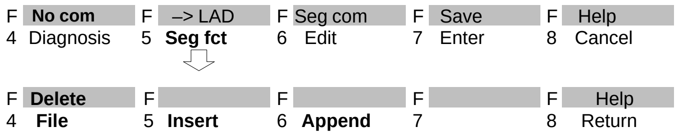
Regardless of the setting “*comments: yes/ no*”, you can switch over between the different displays with ***SHIFT F4*** as follows:

- no comments
- line (statement) comments
- symbol (operand) comments

The setting you select is entered in the settings (page1).

F	**No com**	F	–> LAD	F	Seg com	F	Save	F	Help
4	Diagnosis	5	Seg fct	6	Edit	7	Enter	8	Lib no

Appending, Inserting, Transferring, Deleting a Segment



The segment is in the output mode.

If you want to work with segments in the block, i.e.:

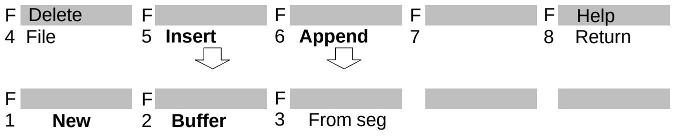
- to append or insert,
- to file (save temporarily)
- to delete,

you can perform these functions using the function keys or the keys in the numeric pad (→ *Appendix, key assignment*).

Segment processing function	Function keys	Key in numeric pad
Save segment temporarily	***F4*** *= File*	–
Insert before current segment	***F5*** *= Insert*	Insert segment
Append after current segment	***F6*** *= Append*	Segment end
Delete segment	***SHIFT F4*** *= Delete*	Delete segment

Appending or Inserting a New Segment

1. Open the segment before or after which you want to add a new segment.
2. Press ***F5*** = *Seg fct.*



3. Press ***F5*** = *Insert* again if you want to insert a segment in front of the current segment or ***F6*** = *Append* if you want to append a segment after the current segment.
4. Press ***F1*** = *New.*

   STEP 5 displays a new segment

Copying a Segment

You can copy a segment within the same block or to a different block in the same program file. The segment title and comment are also copied. After you have copied a segment, it is advisable to update the cross reference list if you have not already selected “Update XRF” in the job box.

Ready to Start?

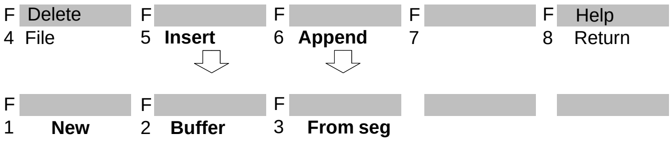
The block to which you want to copy a segment exists in the program file. Copying is done in the output mode.

Copying a Segment in the Same Block

Note

Segments within a function block that contain functions for the specific function block, for example labels cannot be copied to another position within the block.
When you copy a segment, jump labels with symbolically defined names (e.g. MARK) can only be represented in absolute format (e.g. M0001).

1. Open the block before or after the segment to be copied.
2. Press ***F5*** = *Seg fct.*



3. Press ***F5*** = *Insert* again if you want to copy before the current segment or ***F6*** = *Append* if you want to append a segment after the current segment.
4. Press ***F3*** = *From seg.*

   STEP 5 displays the message line “Seg no”.
5. Enter the segment number of the segment to be copied (e.g. 2) and press the ***Return*** key.

The segment is copied.

Copying a Segment to a Different Block

1. Display the segment to be copied using ***page forwards/ backwards***.
2. Press ***F5*** = *Seg fct.*

Filing the Segment

3. Press ***F4*** = *File.*

The segment is temporarily stored.

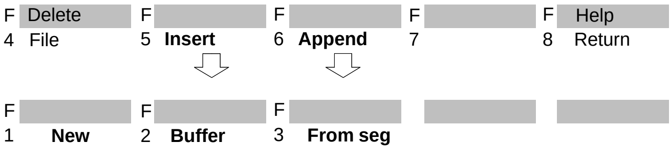
4. Press ***F8*** = *Return.*

Returns you to the block editor in the output mode.

5. Exit the block, assuming you have made no corrections with the ***ESC*** = *Cancel* key, otherwise press the **Insert** key.

Copying the Segment

6. Open the block with the segment before or after which you want to insert the copied segment.
7. Press ***F5*** = *Seg fct.*



8. Press ***F5*** = *Insert* again, if you want to insert in front of the current segment or ***F6*** = *Append* if you want to append the segment after the current segment.
9. Press ***F2*** = *Buffer.*

The buffered segment is copied.

10. Press ***F8*** = *Return.*

Returns you to the block editor in the output mode.

Deleting a Segment

You can delete individual segments in the block. The segment title and comment are also deleted. After deleting a segment, you must update the cross reference list (XRF).

Ready to Start?

The segment to be deleted is open. STEP 5 is in the output mode.

Deleting a Segment in the Block

1. Press ***F5*** = *Seg fct.*
2. Press ***SHIFT F4*** = *Delete* and acknowledge with ***yes*** if you really want to delete the segment.

The segment along with its title and comment is deleted, but not removed from the program file. This only occurs at the end of the editor editing session when you store the block.

3. Press ***F8*** = *Return.*

Returns you to the block editor in the output mode.

Note

With ***SHIFT*** *and* ***delete segment*** in the numeric pad you can also delete a segment.

Transferring/ Shifting a Segment

You can transfer a segment within the same block or to a different block in the same program file. This function is a combination of → *Copying a segment* and → *Deleting a segment*. After the transfer you must update the cross reference list 〈→ *generate XRF*).

How to Transfer a Block

The procedure for transferring segments is the same as for copying segments (→ *Copying a segment to a different block*) with the difference that after you have buffered the segment (file) the segment must be deleted at its old position using ***F4***.

Press ***SHIFT F4*** = *Delete* and confirm with ***yes***.

Creating, Displaying Cross References, Block Change

The cross references of all blocks in a program file are stored in a special program file *XR.INI. You can access this data using the function ***F2*** = *Reference*. With this function, you can do the following:

– Create a cross reference list with ***F1*** = *Gen XRF.*

– Display cross references of an operand on the screen using ***F2*** = *Disp XRF.*
– Trigger a block change by selecting a reference in the cross reference list using the cursor and pressing ***F2*** = *Jump*,
– Change blocks by specifying the destination block and segment using ***F4*** = *Dest blk* and...

if you have changed blocks, you can return to the original block with ***F5*** = *Orig blk*.

You can display a cross reference list of the following operands:

– inputs/outputs
– flags/extended flags
– timers/counters
– block calls
– process I/Os
– data and symbols.

Ready to Start?

STEP 5 is in the output mode. The file XR.INI exists and has been updated. You can achieve this situation as follows:
– by setting → *Update XREF* in the “Edit STEP 5 block(s)” job box; XR.INI is then updated when you stored a block,
– as an alternative, you can use the management function → *Generate XRF.*

Working with the Function “Gen XRF”

With this function, you create the cross reference list for the preset program file with the name *XR.INI:

After you start the function, it is executed automatically.

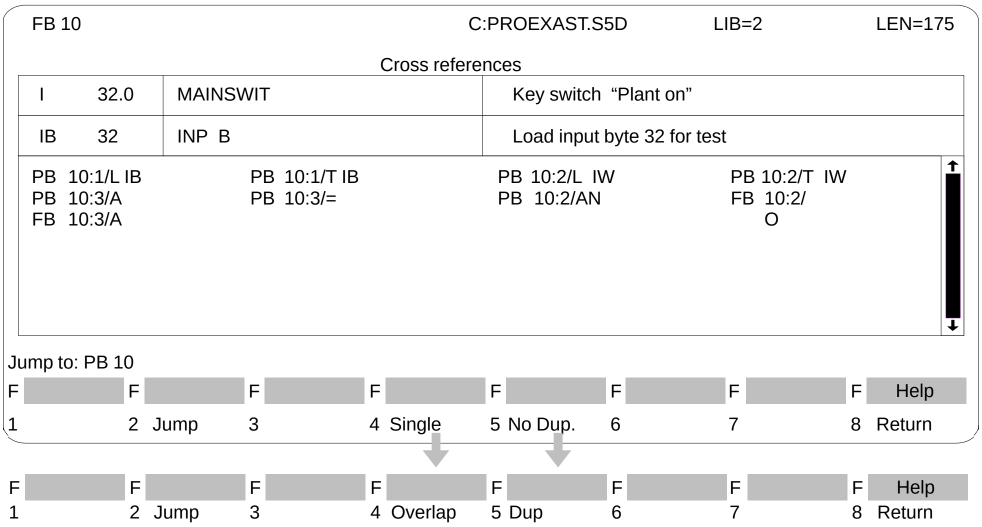
The created cross reference list is required in the block editor for documentation in KOMDOK format and in GRAPH 5 for executing the functions associated with ***F2*** = *Reference*.

Restrictions of “Gen XRF”

When creating the XRF within the editor, there is less memory available than for generating an XRF, that is strated directly from the menu. This means that with large program files, data must be buffered in temporary files earlier. This slows down the generation of the XRF.

Working with the Function “Display XRF”

1. Position the cursor on the statement containing the operand whose cross references you want to display or if the operand does not exist in the current segment, start at step 2.
2. Press ***F2*** = *Reference.*
3. Press ***F2*** = *Disp XRF.*
   STEP 5 displays the message: “XRF display of the operand: e.g. I 32.0”.
4. Enter the operand or overwrite it and press the ***Insert*** or ***Return*** key. The cross reference list of the operand is displayed for example:



```
FB 10                                  C:PROEXAST.S5D         LIB=2            LEN=175
                                     Cross references
```

I 32.0	MAINSWIT	Key switch “Plant on”
IB 32	INP B	Load input byte 32 for test

```
PB  10:1/L IB          PB  10:1/T IB          PB  10:2/L  IW          PB 10:2/T  IW
PB  10:3/A             PB  10:3/=             PB  10:2/AN             FB  10:2/
FB  10:3/A                                                                 O
```

Jump to: PB 10

5. ***F4** = Overlap/Single:*
   ”Overlap”: the cross reference list also contains the byte, word or double word addresses that overlap the bit or byte address of the displayed operands.
   ”Single”: only cross references of the specified operand. If the cross reference list is long or if you do not have enough memory, overlapping can be switched off.
6. ***F5**= Dup/No dup:*
   ”Dup”: if an operand occurs with the same operator more than once in a segment, it is displayed as often as it occurs.
   ”No dup”: the operand with the same operator in a segment is only displayed once. This setting is advisable in long cross reference lists and when you do not have a lot of memory.
7. You can return to the previous level with ***F8** =,Return* or ***ESC**.* You can jump to a different block using ***F2** = Jump.*

Select help (***SHIFT F8***) and reply ***yes*** to the “***Continue?***” prompt, you will obtain detailed information about the functions.

Jumping to a Block

1. With the cursor in the cross reference list, select the block you want to change to.
2. Press ***F2** = Jump.*

The selected block is displayed.

3. You can return to the previous segment with:

   ***F8** = Return*
   ***F5** = Orig blk.*

Changing Blocks

1. Press ***F2** = Reference.*
2. Press ***F4** = Dest blk.*
   STEP 5 displays “Jump to block: segment: 1”
3. Type in the block and overwrite the segment number if you require a different one.
4. Press the ***Insert*** key.

The selected block is displayed.

Searching for Operands

Using the search function you can find certain terms, for example operands, quickly in the open block. The key is searched for from the cursor position or from the first segment. If STEP 5 finds the key, this is displayed in the corresponding segment.

What can you search for ?

– Absolute operands	I, F, S, Q, T, C
– Block calls	OBn, PBn, SBn, FBn, FXn, DBn, DXn
– Peripheral bytes/words	PYn, PWn
– Data	DRn, DLn, DWn, DDn, Dn.m
– Symbolic operands	e.g. -INPUT
– Assignment for absolute or symbolic operands	
e.g.* Q1.0,	* –INPUT

Ready to Start ?

STEP 5 is in the output mode.

How to Search for a Key

Searching in the block

1. Press ***F3*** *= Search.*
2. Type in the key in absolute or symbolic form, e.g. **I 1.1** .
3. Start the search, as follows:
   from the 1st segment - press ***F2*** *= From seg1* or
   from the next statement line - press ***F3*** *= Continue*.

Continuing the search

Press ***F3*** *= Search*, see above.

Editing Symbolic Operands in the Block

Symbolic operands can be edited in a list directly in the block. This list is an excerpt from the symbols file *Z0.INI and the operands of the open segment are displayed.

If you change anything, the sequential source file *Z0.SEQ should be updated as follows:

- by setting → *Update SEQ. source file,* in the “STEP 5 block” job box, so that the *Z0.SEQ is updated when the block is stored,
- or you can generate the sequential source file from the symbols file (→ *Management, Assignment lists, Convert INI > SEQ*).

Ready to Start ?

You have selected “symbols: yes” in the “settings” (→ *Project*). If this is not the case, you can switch over with ***SHIFT F3***.

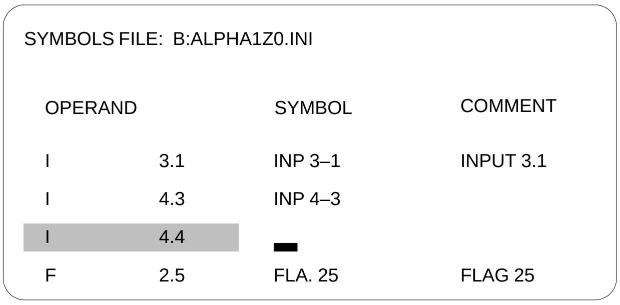
How to Edit Symbolic Operands

1. Press ***F1*** = *Disp symb.*

A list containing the operands is displayed on the screen.

2. Select the operand with the long cursor.
3. Press ***F2*** = *Edit symb.*

The character cursor is located in the symbols column.



```
SYMBOLS FILE:  B:ALPHA1Z0.INI

OPERAND               SYMBOL            COMMENT

I          3.1        INP 3–1           INPUT 3.1
I          4.3        INP 4–3
I          4.4        ▬
F          2.5        FLA. 25           FLAG 25
```

4. Enter the symbolic name keeping to the original upper and lower case letters.
5. Position the cursor in the comment column with ***SHIFT*** and the ***cursor right*** key or with the ***Return*** key.
6. Type in the comment keeping to the upper and lower case letters of the original.
7. Complete the edited line by pressing ***F2*** = *Insert.*
8. To complete editing, press ***F8*** = *Return* and the ***Insert*** key.

Note

Symbolic names should begin without a hyphen. Do not use umlauts (ä, ü, ö).

3.3.2 Editing Statement Lists

Editor
STEP 5 block
in the program file
in the PLC...

The STEP 5 statement is the smallest independent unit of a program. It represents a task description for the processor. In the “Statement List” (STL) method of representation, a statement is typed in per line in either absolute or symbolic form (possible blocks: OB, PB, SB, FB/FX). A statement consists of the operand and the operand as follows:

	Operation	Operand
Absolute statement	AN	I 1.1
Symbolic statement	AN	–INPUT

You can write a maximum of 255 statements per segment.

Note

For an example of editing a Statement List, see Chapter 2.

Ready to Start ?

If you are using symbolic operands, a symbols file must exist and the name of the symbols file must be entered in the settings (page 1) and “symbols: yes” selected in page 2. If this is not the case, you can switch over with ***SHIFT F3***.

Statements are always entered in the “edit” mode. If you call a new block, STEP 5 is in the edit mode, if you call an existing block, STEP 5 is in the output mode. In this case, you must switch over to the editing mode with ***F6*** = *Edit*.

Typing in Statements

When typing in a statement, you do not have to keep to the strict format, i.e. STEP 5 enters the blanks automatically after you have entered the line. Complete each line with the ***Return*** key. The cursor is positioned in the first line.

Type in the first statement or position the cursor on the required line and type in the statement, e.g. ***AN I 1.1*** or ***AN–INPUT*** and press the ***Return*** key.

Correcting Statements

Position the cursor on the statement and overwrite. You can delete individual characters with the ***DEL*** key.

Storing the Block

Press the ***Insert*** key.

STEP 5 switches to the output mode.

Press the ***Insert*** key again.

Displaying Addresses

With this function you can display the relative operation addresses in bytes or words when editing in STL. While the addresses are displayed, you cannot edit statements and cannot enter a library number.

Ready to Start ?

STEP 5 is in the output mode.

How to Display Addresses

1. Press ***SHIFT F1*** = *Addresses.*

   STEP 5 displays the relative addresses in words.
2. Press ***SHIFT F1*** = *Addresses* again.

   STEP 5 displays the relative addresses in bytes.
3. Press ***SHIFT F1*** = *Addresses.*

   STEP 5 returns to the Statement List without addresses.

Note

If you display the addresses from the PLC online, they are only displayed in words. If you press ***SHIFT F1*** = *Addresses* a second time, the address information is cleared from the screen.

Statement Comment

Statement comments are stored in comment blocks just as → *Segment titles*. While the input of segment titles is not dependent on the method of representation, you can only assign a (line) comment to a single statement in the STL editor. A statement comment has a maximum of 32 characters.

You can type in a statement comment directly when programming the statement without having to open the comment block in the program file. In this case, the comment block is generated automatically when you store the STEP 5 block.

You can also enter statement comments separately in the comment block. We recommend the first method, since the comment block is automatically updated if you make any changes. The names of the comment blocks are assigned automatically by STEP 5 as follows:

OCn for OBn,
PCn for PBn,
SCn for SBn,
FCn for FBn,
FCXn for FXn.

Ready to Start ?

You have selected “*Comments: yes*” in the settings (→ *Project*).
If this is not the case, you can switch over with ***SHIFT F4***.

STEP 5 is in the edit mode.

How to Enter Statement Comments

1. Position the cursor on the required statement.
2. Move the cursor to the right to the comment field *(**TAB** and **cursor right**).*
3. Type in a text with a maximum of 32 characters or correct an existing text.

After the 32nd character, the cursor jumps to the beginning of the comment field.

4. Press the ***Return*** key.

How to Store the Comment

The first time you store the block with comments, the comment block (OC, PC, SC, FC/FCX) is generated automatically.

If the comment block already exists, STEP 5 displays the message: “Enter modified block?”.
Enter the comment with the ***Insert*** key or discard it with ***ESC*** = *Cancel*.

Function Block

A function block (FB, FX) is a STEP 5 program block similar to OBs, PBs and SBs. While these blocks only contain the basic STEP 5 operations, an FB or FX can contain the following:

- basic operations,
- supplementary operations and
- system operations.

An FB occurs only once in the program memory of the programmable controller. When you program the block, you decide on its function, and the operands you enter can be formal operands which have a token function. When the block is called (→ *Calling a function block*) the higher ranking block replaces the formal operands by actual operands.

Structure of an FB

A function block consists of the following:

- a block preheader (FV, FXV),
- a block header (as with all other blocks),
- a block body (as with all other blocks).

Block Preheader

The block preheader contains the identifiers of the jump labels that you have entered in the block. The block header is

- automatically generated by STEP 5 when the block is translated,
- stored in the preset program file as an FV or FXV,
- not transferred to the PLC and not to EPROM/EEPROMs,
- automatically deleted when its FB or FX is deleted.

If the block preheader does not exist when a function block is transferred from the PLC memory to the selected program file, STEP 5 displays the following message: “Preheader does not exist for this block”.

Block Header

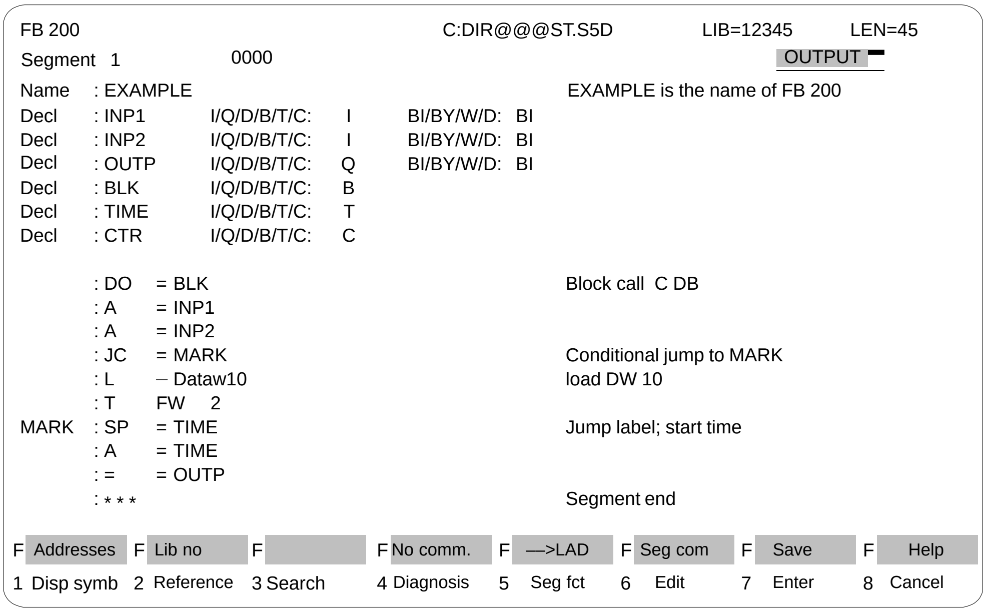
The block header contains the following:

- the block type and block number,
- the library number,
- the block length.

Block Body

The block body contains the STEP 5 program and a parameter list with the block parameters of all segments of the function block. This parameter list contains all the information necessary to perform the following tasks:

- to represent the block graphically (e.g. input, output parameters),
- to check that the data type is entered correctly when the actual operands are input (parameter assignment).



```
FB 200                                          C:DIR@@@ST.S5D        LIB=12345      LEN=45
Segment  1             0000                                                   OUTPUT
Name    : EXAMPLE                                      EXAMPLE is the name of FB 200
Decl    : INP1       I/Q/D/B/T/C:    I       BI/BY/W/D:  BI
Decl    : INP2       I/Q/D/B/T/C:    I       BI/BY/W/D:  BI
Decl    : OUTP       I/Q/D/B/T/C:    Q       BI/BY/W/D:  BI
Decl    : BLK        I/Q/D/B/T/C:    B
Decl    : TIME       I/Q/D/B/T/C:    T
Decl    : CTR        I/Q/D/B/T/C:    C

        : DO   = BLK                                   Block call  C DB
        : A    = INP1
        : A    = INP2
        : JC   = MARK                                  Conditional jump to MARK
        : L    – Dataw10                               load DW 10
        : T    FW    2
MARK    : SP   = TIME                                  Jump label; start time
        : A    = TIME
        : =    = OUTP
        : * * *                                        Segment end
```

Figure 3-19 Example of a Function Block (FB 200)

Editing a Function Block

A function block can contain the STEP 5 statements, a block name and a parameter list of the formal operands. Jumps or branches can be programmed within a segment.

- Programming is also possible in LAD and CSF. Except for the first segment, all the new language elements can be used within a segment (→ *Editor, LAD/CSF*)
- The formal operands defined in the first segment cannot be used in a LAD or CSF segment.
- The FB name is displayed in the “Directory” function (→ *Object*, *Blocks*)

”Name”

The block name can be up to 8 characters long and must begin with a letter.

Parameter List

The parameter list contains the name, the parameter type and the data type of the formal operand. You can include a maximum of 40 formal operands per function block.

”Decl”

Name of the formal operand, with a maximum of 4 characters, the first of which must be a letter.

”I/Q/D/B/T/C”

The type of formal operand:

I	Input parameter
Q	Output parameter
D	Data (constant)
B	Block call (C DBn/DXn, JU OBn, PBn, SBn, FBn/FXn)
T	Timer
C	Counter

”BI/BY/W/D”

Type of data:

BI	Operand with bit address
BY	Operand with byte address
W	Operand with word address
D	Operand with double word address

Editing a new function block

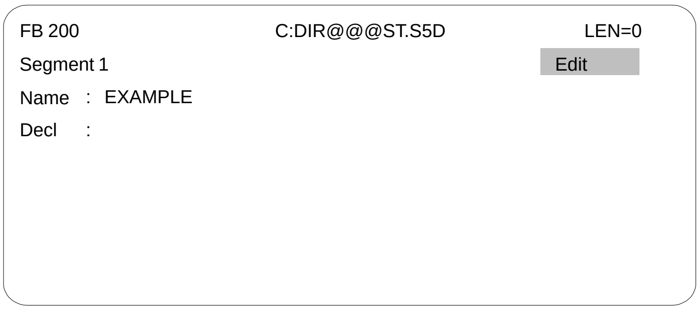
Ready to Start ?

For symbolic operands, a symbols file must exist and “*symbols: yes*” must be selected in the settings. STEP 5 is in the edit mode (STL). Segment 1 is open.

How to Input an FB

1. Type in a name with a maximum of 8 characters, e.g. ***EXAMPLE1***
   If the name is 8 characters long, the cursor jumps to the comment file (→ *Statement comment*).
2. Press the ***Return*** key.

The parameter list for the formal operands is opened and “Decl:” is displayed.



```
FB 200                       C:DIR@@@ST.S5D                   LEN=0
Segment 1                                                   Edit
Name  :  EXAMPLE
Decl  :
```

3. Type in a maximum 4 character string for the first formal operand.

   After 4 characters, the cursor jumps to the next input field. If you use less than 4 characters, jump to the next field with the ***Return*** key.
4. Select the type of formal operand, e.g. type in ***I***

   The cursor jumps to the next input field.
5. Select the type of data, e.g. type in ***BI***

If you only use one character here, press the ***Return*** key. The cursor jumps to the next line in the parameter list.

6. Continue to enter the parameters as described above.
7. Complete the parameter list by pressing the ***Return*** key.

The cursor jumps to the first line of the block body, where you can enter the first statement.

Note

You can only add parameters later between the lines of the parameter list if the parameters already entered do not yet exist in any statement line in the block body.

Calling a Function Block

When you call the function block, the actual operands are assigned to the formal operands.

Ready to Start ?

STEP 5 is in the edit mode. The function block to be called is in the program file.

How to Assign Parameters

1. Type in the block call, as follows:

JU FB	for an unconditional FB call
JC FB	for a conditional FB call
DO FX	for an unconditional extended function block *call*
DOC FX	for a conditional extended function block call

2. Press the ***Return*** key.

   The PG displays the name of the FB.
3. Press the ***Return*** key.

   In the next line, STEP 5 displays the first formal operand and waits for you to type in the first actual operand.
4. Type in the actual operand in absolute or symbolic form and press the ***Return*** key.

Note

Absolute actual operands for BI, T, C must be entered with a blank, for example I 1.0.

5. Type in the remaining actual operands and complete each one with the ***Return*** key.

STEP 5 sets the type of parameter and data type automatically which you can either accept or change.

1. Press the ***Return*** key in the line of the formal operand or move the cursor to the right.

   STEP 5 displays the type of parameter you selected in the parameter list.
2. Either accept the displayed parameter type or overwrite it with a different type.
3. Press the ***Return*** key.

3.3.3 Editing Ladder Diagrams

Editor
STEP 5 block
in the program file

In the Ladder Diagram method of representation (LAD) the control task is described based on the symbols used in circuit diagrams. Based on these symbols, the block operations are represented by contacts (NC contacts, NO contacts, outputs) and function symbols for counters, timers and arithmetic operations. You can program in LAD in the following STEP 5 blocks:

- organization block OB
- program block PB
- sequence block SB
- function block FB
- extended function block FX.

STEP 5 stores the corresponding segment comments in the blocks OBDO.nnn, PBDO.nnn etc. Segment titles are stored in the comment blocks OC, PC etc.

It is advisable to enter and correct comments when editing a block and not to write them directly in the documentation or comment blocks.

New Functions

Compared with previous STEP 5 versions, the following functions are now available when editing in LAD:

1. The softkey menu has been extended from 8 to 16 function keys and can be activated with the mouse.
2. Function blocks (FB, FX) can be created, however, the first segment must still be programmed in STL.
3. The first segment must be programmed in STL.
4. No formal operands can be used.
5. The following extended functions can be invoked using softkeys:
   - conditional and unconditional block calls for OB, PB, SB, DB, DX, FB, FX
   - loading and transfer operations
   - negated connector
   - SHIFT and rotate operations (only in FB, FX)
   - conversion operations (only in FB, FX)
   - digital logic operations (only in FB, FX)

Working with the LAD Editor

Settings for editing → ***Object, Settings, Page 1 and 2*** and then

Program file:	Name of the currently selected program file.
Symbols file:	If you want to work with symbolic operands, you must specify the symbols file, select “symbols: yes” on page 2 and specify the symbol length.
Mode:	”Online” if a PLC is connected and you want to edit in the PLC.
Representation:	”LAD”
Comments:	“Yes” and the comment length (max. 40 characters) if you want to edit segment titles and comments.

If you are editing existing blocks, you can select LAD with ***SHIFT F5*** = *LAD* regardless of the selected method of representation.

Calling the Editor

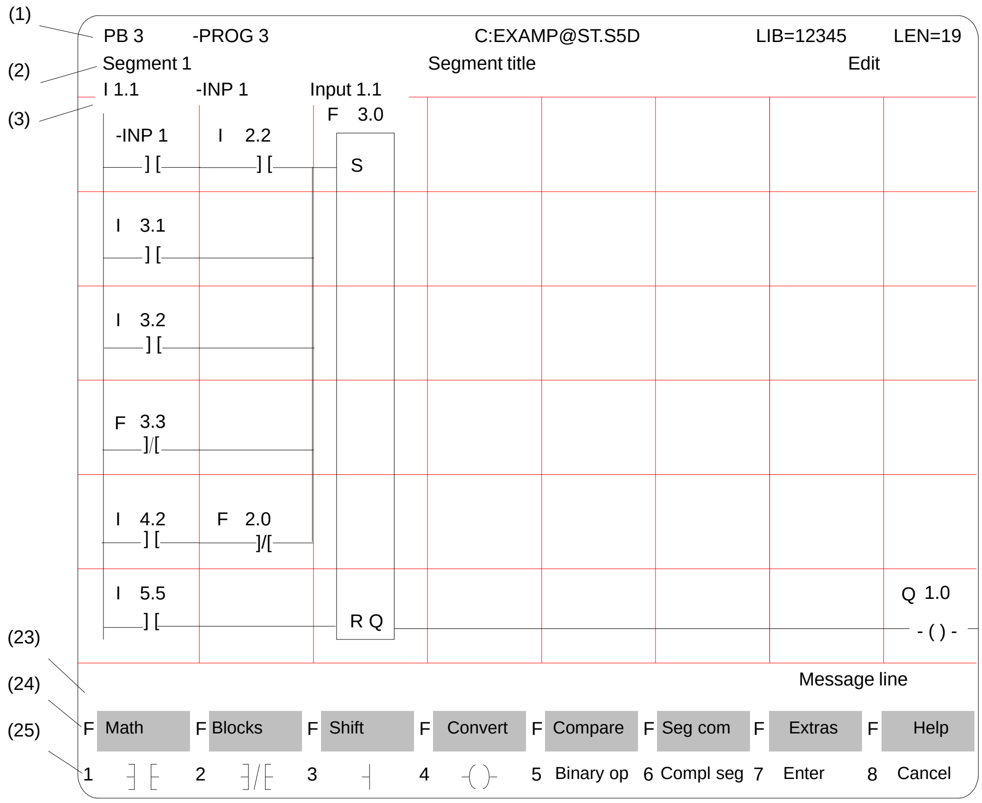
After selecting the editor function “Edit STEP 5 blocks in the program file” or “in the PLC” the job box is displayed. Here, you name the block and it is advisable to select the options “Update XRF” and “Update seq. source file” if you are working with symbolic operands.

After confirming your entries with ***OK***, the Ladder Diagram editor is called. A working field appears on the screen (*Figure* 3-20) and the softkey menu with the symbols for entering contacts and processing LAD segments.
The editing field is divided into lines and columns in which you enter rungs, branches, contacts, outputs and function elements using the keys of the softkey menu or the mouse. As you build up your segment, you are supported intensively by STEP 5. Connections and symbols of all types (e.g. signal inputs/outputs for counter or arithmetic functions) are generated automatically. Input fields for labelling and assigning parameters are displayed and can be reached with the automatic cursor control. STEP 5 rejects inconsistent configurations.

Screen Layout

The screen is divided into 48 fields (8 columns and 6 horizontal sections). The horizontal sections are 3 lines high. The first 7 columns contain logic operations, the 8th column is reserved for the outputs.

The label and the corresponding contact are arranged one above the other in one of the 48 fields.
The Ladder Diagram can be a maximum of 2.5 times the length of one screen.



PB 3 -PROG 3 C:EXAMP@ST.S5D LIB=12345 LEN=19

Segment 1 Segment title Edit

I 1.1 -INP 1 Input 1.1

Message line

Figure 3-20 Segment in Ladder Diagram Representation (Example)

Table 3-1 Explanation of the Screen Lines (*Figure* 3-20)

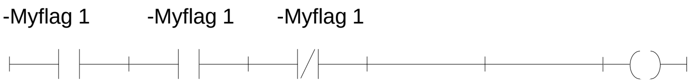
Line	**Display**	**Explanation**
(1)	PB3 –PROG3 C:EXAMP@ST.S5D LIB=12345 LEN=19	Block type and number Symbolic block name Drive and program file Library number Block length in words
(2)	Segment 1 Segment title Edit	Segment number Text with max. 32 characters STEP 5 mode
(3)	Symbolic operands	Assignment “absolute operand” → “symb. operand” → operand comment, when the cursor is located on an operand identifier
(4)... (22)	Editing area	Input fields for logic operations, calls and operands
(23)	Message line	STEP 5 messages or prompts (red or on a black background)
(24) (25)	Function keys	Key assignment for the currently active functions

Naming Operands

After you input a LAD symbol, the cursor jumps to the name field (max. 8 characters) for the operand. If you have selected a symbol length greater than 8 characters in the “Settings”, STEP 5 only displays the first 8 characters. If you use longer symbol names, make sure that the names are unique within the first 8 characters. Example: you have the following assignment:

Operand	Symbol	Comment
F 100.1	Myflag 100	
F 1.1	Myflag 1.1	
F 1.7	Myflag 1.7	

The selected symbolic operand names are displayed or printed out as follows:



There are two methods of naming operands, as follows:

1. The operand can be named immediately after selecting a symbol (automatic cursor positioning),
   or after exiting the name field [?????] with the ***Return*** key.
2. Entering the operand names in the name fields of the completed segment, guided by the long cursor.

Note

You can only exit a segment or block when all the names and parameters have been input correctly.

Reconfiguring a Segment

If while editing a segment its layout has become awkward (e.g. as a result of repeated branches) you can redisplay the segment by pressing ***SHIFT F7*** = *Extras* and ***F2*** = *New disp* even if the segment does not yet have all the required parameters. The screen is then reconfigured and the display layout optimized.

Editing Symbolic Operands

When you press ***F1*** = *Disp symb* in the output mode, STEP 5 displays a list of operands in absolute and symbolic form for the open segment. You can then edit this list. If you use longer symbol names make sure that the names are unique within the first 8 characters. The symbolic operand names are reduced to 8 characters on the screen and when printing in LAD and CSF.

If you make changes, it is advisable to update the sequential source file if you have not already selected this function in the job box.

Logic Operations

When you select “LAD” on page 2 of the “Settings” and “Edit STEP 5 block(s) in ...”, STEP 5 opens the block selected in the job box at segment 1. If you are working with a new block, this is empty apart from the power rail on the left-hand side.

Using the function keys, you can now input contacts, outputs and function elements (*Tables 3-1 and 3-2*). The left-hand column of these tables contains the operation for processing the contact(s) which you call in the edit mode using the keystrokes shown.

Table 3-2 Logic Operations in LAD (Ladder Diagram)

Operation	Softkeys	Explanation
—| |—	***F1***	NO contact
—|/|—	***F2***	NC contact
——|	***F3***	Branch, close branch
—( )—	***F4***	Output
Binary op	***F5***	Call complex functions
—(#)—	***F5+F4***	Connector
—(/)—	***F5+F5***	Negated connector
——	[→] **(Cursor right)**	Empty element

Editing Series and Parallel Rungs

When you input the first contact at the position marked by the long cursor in the empty segment, you generate a continuous rung including the output symbol. You can include up to 7 contacts in series within this rung by positioning the long cursor on the empty element and selecting the required function (Table 3-2).

Further parallel circuits are connected to this continuous rung. A parallel circuit must be continued as far as the close branch point, if necessary by inserting empty elements. Only then is it possible to label elements or make corrections.

You can always connect a parallel circuit to the power rail. Branches can be generated by positioning the long cursor below a contact. The branch point is then generated before this contact. You select the close branch point if necessary by including empty elements using ***F3*** = *Close branch.*

If you attempt to branch from an empty element, this is rejected with the message “parallel circuit illegal”.

Inserting Contacts

You can always insert a contact where there is an empty element. Before you can insert a contact in a rung, you must first expand the rung with ***SHIFT F7*** = *Extras*, ***F6*** = *Hor exp.* or the ***expand (horizontal)*** *key.*

Series

Position the long cursor on the contact after the insertion point and press ***SHIFT F7*** *= Extras and* ***F6*** *= Hor exp.*

All the lines of the segment are moved one column to the right.

Now position the long cursor on the inserted empty element and insert the contact using ***F1*** *or* ***F2*** *or the connector with* ***F5*** *= Binary op +* ***F4*** *= # or* ***F5*** *= /.*

When you store the segment (***Insert***) or reconfigure the screen (***half screen***) unnecessary empty elements are discarded.

Parallel

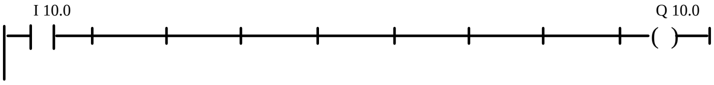
You can generate parallel circuits within a segment as described above by positioning the long cursor between the paths below the contact in front of which you want to start a parallel circuit.

1. Select the required contact with ***F1*** ... ***F4*** .

STEP 5 now expands your segment implicitly without you pressing ***SHIFT F7*** = *Extras*, ***F7*** = *Vert exp* or the ***expand (vertical)*** key and makes room for a new parallel path.

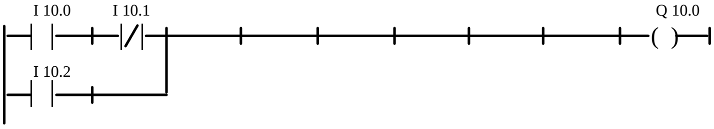
Examples of Editing Logic Operations

Initial display after pressing ***F1*** = *NO contact* and entering the operand identifier ***I 10.0*** and pressing the ***Return*** key and ***Q 10.0*** for output and pressing the ***Return*** key.
Initial situation:



Series and Parallel Contact

1. Series contact: position the cursor in the second column and press ***F2*** = *NC contact* type in ***I 10.2*** and press the ***Return*** key.
2. Parallel contact: position the cursor below the contact **I 10.0** and press ***F1*** = *NO contact.* The parallel branch is opened. Then move the cursor to the right , press ***F3*** = *Close branch,* type in ***I 10.2*** and press the ***Return*** key.

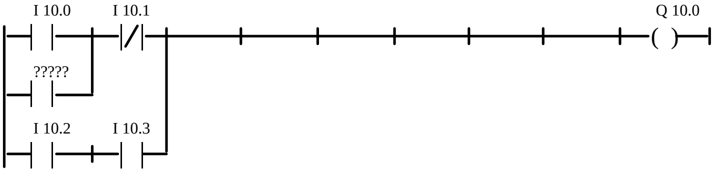


Implicit Expanding

3. Inserting an NO contact in a further parallel branch: position the long cursor below contact I 10.0 once again and press ***F1*** = *NO contact* and ***F3*** = *Close branch.*

Replacing an Empty Element by a Contact

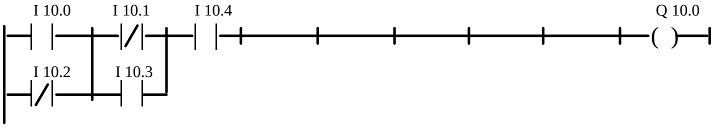
4. Contact I 10.3 is generated by positioning the cursor on the empty element and pressing ***F2*** = *NC contact.*



Bridge Circuit

You can obtain the bridge circuit below as follows:

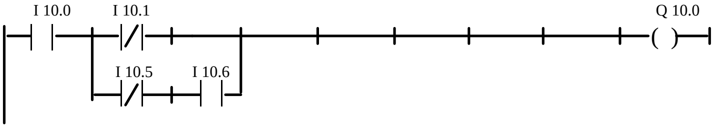
1. In the upper rung: position the cursor on the second column and press ***F2*** *= NC contact* then position the cursor in the third column and press ***F1*** *= NO contact.*
2. Creating the parallel branch: position the cursor below contact I 10.0, press ***F2*** *= NC contact* and ***F3*** *= Close branch* and position the cursor in the second column of the parallel branch, press ***F1*** *= NO contact* and ***F3*** *= Close branch.*



Opening a Branch after a Contact

The following segment shows a parallel path opened after the first contact.

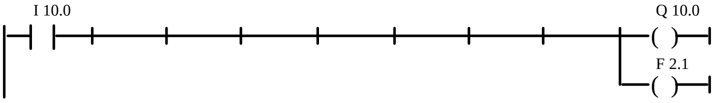
1. In the upper rung, position the cursor on the second column and press ***F2*** *= NC contact* for I 10.1.
2. Creating the parallel branch: position the cursor below contact I 10.1, press ***F2*** *= NC contact,* ***F1*** *= NO contact* and ***F3*** *= Close branch.*



Assignment

Connecting an output or an assignment:

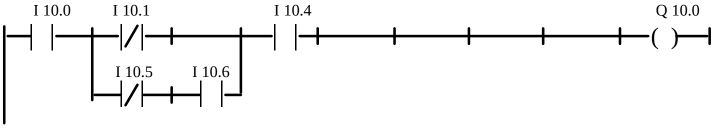
Position the long cursor under output Q 10.0 and press ***F4*** *= Output.*



Using Connectors

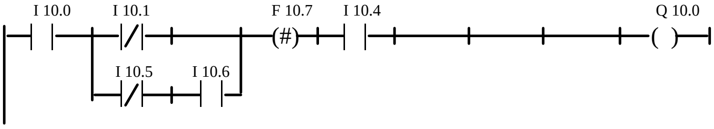
Connectors and negated connectors (Table 3-2) are intermediate flags in binary logic operations. They store the RLO formed up to that point. A connector is input in LAD in the same way as a contact. If it is located after the last contact of the rung it is represented as an output after the rung is entered and stored.

Immediately after the parallel branch is closed, the intermediate RLO is written to flag F 10.7.



Since it is not possible to expand the rung horizontally at this point, contact I 10.4 must first be deleted and then inserted again after the connector, as follows:

1. Position the cursor on the contact below I 10.4 and press ***DEL***.
2. Now position the cursor on the empty element and press ***F5*** = *Binary op* and ***F4*** *= Connector* to create a connector which you can then label ***F 10.7***. Following this, insert contact ***I 10.4*** again.



Complex Functions

In LAD, the following rules apply to the complex functions listed in Table 3-2:

1. All operations (1) to (10) in Table 3-3 are represented as “long boxes” in which the operands are displayed on the left before the processing and on the right the result of the processing. STEP 5 enters the operation selected with the softkeys in the long box itself.
2. Only one complex function is possible in a segment, i.e. a new segment must always be opened.
3. Some function elements can be extended, i.e. the number of inputs can be increased provided the operation allows. To do this, position the cursor on the “roof” of the box and press the vertical expand key.
4. The “shift/rotate” function (4) requires the shift parameter “n” to be entered in the long box, i.e. the number of bits by which the content of the ACCU is shifted left or right. The maximum possible shift depends on the format of the operand (16 or 32 bits).
5. With the functions “Math” and “Compare” you can specify a different operand type in the long box. The type “fixed point number = F” is the default.

Note

The type can only be changed once directly after calling the long box.

In the editing mode, the following functions can be called with ***SHIFT*** or ***F5***: = *Binary op*:

(1) Arithmetic operations
(2) Block calls
(3) Load and transfer operations
(4) Shift and rotate word/double word operands
(5) Binary operations
(6) Conversion operations
(7) Comparison between two operands
(8) Digital logic operations
(9/10) Counter and timer operations

Table 3-3 Complex Functions in LAD

Operation	Keys (softkeys)		Explanation
Math. ADD, SUB MULT, DIV	***SHIFT F1*** and	***F1, F2*** ***F3, F4***	(1) **Arithmetic operations:** Addition, subtraction multiplication, division
(with FBs/FXs) AND OR XOR	**SHIFT F1** and	**F5** **F6** **F7**	*(8)* **Digital logic operations:** AND operation, words OR operation, words Exclusive OR operation, words
Blocks JU FB, JC FB DO FX, DOC FX JU..., JC... C DB, CX DX	***SHIFT F2*** and	***F1, SHIFT*** **F1 F2,** ***SHIFT*** **F2 F4,** ***SHIFT*** **F4 F6,** ***SHIFT*** **F6**	(2) **Call blocks as follows:** FB unconditional, FB conditional FX unconditional, FX conditional OB, PB, SB unconditional, conditional DB, DX
(SHIFT) L/T	***SHIFT F3*** and	***F7***	*(3)* **Load and transfer operations** Load and transfer operand

Table 3-3 Complex Functions in LAD

Operation	Keys (softkeys)		Explanation
SHIFT (with FBs/FXs) SLW, SLD SRW SSW, SSD RLD, RRD	**SHIFT F3** and	**F1, SHIFT F1 F2, F3, SHIFT F3 SHIFT F4 SHIFT F5**	(4) **SHIFT and rotate operations** SHIFT word/double word left SHIFT word right SHIFT word/double word with sign right Rotate left, right
Convert (with FBs/FXs) DEF, CFW DUF, CSW DED, CSD DUD FDG, GFD	**SHIFT F4** and	**F1, SHIFT F1 F2, SHIFT F2 F3, SHIFT F3 F4 F5, F6**	(6) **Convert operations** BCD->binary, form 1’s compl., 16 bit Binary->BCD, form 2’s compl., 16 bit BCD->binary, form 2’s compl., 32 bit Binary->BCD, 32 bit fixed point -> floating point, floating point -> fixed point, 32 bit
Compare ! = > < > = < = > <	**SHIFT F5** and	**F1, F2 F3, F5 F4, F6**	*(7)* **Comparator operations** (between two operands): Compare for “equal to”, **“**not equal to” Compare for greater than or equal to, less than or equal to Compare for “greater than”, “less than
Binary op CD, CU	**F5** and	**F1, F2**	(9) **Counter operations:** counter value incremented, decremented by 1
Binary op SP, SE SD, SF SS	**F5** and	**SHIFT F1/F2 SHIFT F3/F5 SHIFT F4**	(10) **Timer operations:** Start timer as pulse, extended pulse Start timer as ON/OFF delay Start timer as stored on delay
R/S S/R	**F5** and	**SHIFT F6 SHIFT F7**	*(5)* **Binary latching operations:** Priority setting flip-flop Priority resetting flip-flop

Arithmetic Operations

(1) in Table 3-3
The operators ADD, SUB, MULT and DIV combine two operands in ACCU 1 and 2 to form a result in ACCU 1. The function corresponds to the following STL statements:

– load operand 1
– load operand 2
– execute the selected logic operation
– transfer result to operand (ACCU 1)

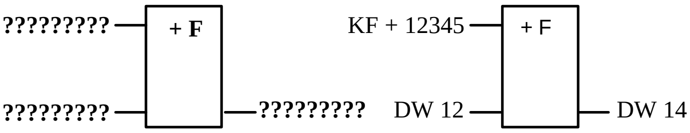
Operand types : KF, DW, IW...

Example

Editing an ADD operation for two fixed point numbers:

1. Press ******* or ***F6*** *= Compl seg* and then ***SHIFT F1*** *= Math.*
2. Select the required operation, here ***F1*** *= ADD.*

STEP 5 displays the long box with undefined inputs and outputs and the default operand format “F”.



3. Confirm the operand format by pressing the ***Return*** key.
4. Type in the 1st operand, in this case KF + 12345 and press the ***Return*** key.
5. *Type in the 3rd operand, in this case DW 12 and press the* ***Return*** *key.*
6. Name the operand to which the result will be transferred (DW 14) and press the ***Return*** key.

The segment now appears as shown on the right-hand side of the figure.

Block Calls

(2) in Table 3-3
Using the STEP 5 block calls with which other blocks in the user program can be called from any block allows structured programming. A block call is represented in LAD either as an output (assignment) or as a long box when calling a function block (FB/FX).

In an empty segment, you can input a call directly using the softkeys. In existing segments, you can insert and append calls with/without implicit expanding of the rung.

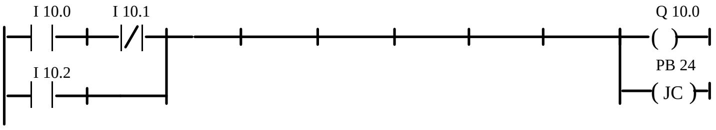
Note

A LAD segment contains either only an unconditional block call or a logic operation with a conditional block call. For this reason, if you press ***F4*** = *Output* the default “JU” or “ = ” (assignment) is displayed.

Example 1

Conditional program block call

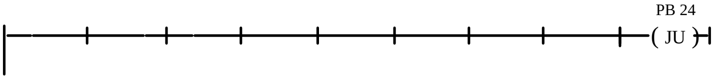
1. Position the cursor below the output symbol and press ***SHIFT F2*** = *Blocks* and ***SHIFT F4*** = *JC ....*
2. Enter the destination block, in this case PB 24, in the input field above the call symbol and complete the entry with the ***Return*** key.



Example 2

Unconditional program block call

Press ***SHIFT F2*** = *Blocks* and ***F4*** = *JU ....*



Example 3

Unconditional FB call in an empty segment

1. Press ***SHIFT F2*** = *Blocks* and ***F1*** = *JU FB.*

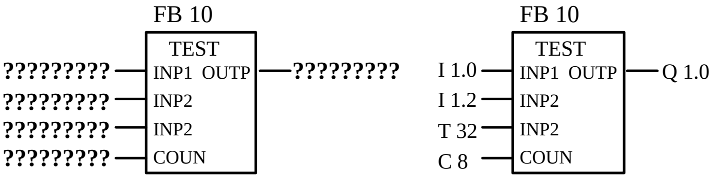
The editor displays the “roof” of the block with the cursor in the labelling field.

2. Type in the name of the function block to be called, in this case ***FB 10***.

The function block with its formal operands is displayed in the form shown on the left-hand side.

3. The cursor is positioned on the input field of the first actual operand. Now type in the operand in absolute or symbolic form. Move to the other fields using the ***Return*** key.

The segment then appears as shown on the right-hand side.



Load and Transfer Operations

(3) in Table 3-3
The function is displayed as a “long box” with the operand to the left and the result to the right. The function ***SHIFT F3*** = *Shift* and ***F7*** = *L/T* correspond to the following STL statements:

- load operand (DW, DD, IW...),
- transfer to operand (DW, DD, IW...).

After generating the long box (see above) you simply enter the operands displayed as [?????].

Shift and Rotate Operations

(4) in Table 3-3
Shift and rotate operations belong to the supplementary operations (only FB, FX). A shift/rotate operation is displayed in an empty segment as a long box with the operand in ACCU 1 to the left before the shift operation and the result to the right. After pressing the softkeys ***SHIFT F3*** = *Shift* and the required function at the second key level, STEP 5 generates the “undefined” long box in which you enter the required operation.

The character cursor flashes below the parameter “n”. Here, you enter the number of bits by which the content of the operand will be shifted.
The function corresponds to the STL statements:

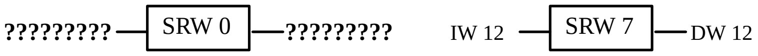
- load operand
- shift/rotate operand by “n” bits
- transfer result to operand (ACCU 1).

Example

Shifting the input operand IW 12 seven bits to the right and transferring to DW 12.

1. Press ****** or F6*** *= Compl seg* followed by ***SHIFT F3*** *= Shift.*
2. Select the required operation, in this case ***F1*** *= SRW.*

STEP 5 displays the long box (left).



3. Position the cursor on the parameter “n” in the box, in this case 0, and type in the number “7”.
4. Type in the input and output operands.

Note

It is not possible to modify “n” at a later time.

Latching Operations

(5) in Table 3-3
Using the latching functions, the RLO can be stored. You can specify how the latching function works after pressing ***F5*** *= Binary op* and then selecting either ***F6*** “priority set” or ***F7*** “priority reset” at the second key level. STEP 5 enters the operands with priority at the top of the long box.
The latching function is displayed as a box with 2 inputs and 1 output, S is the “set” input, R is the “reset” input and Q is the output. Only one latching function can be inserted in a segment.

The latching function corresponds to the following statements (STL):

– A (N)	1st input operand
– S (R)	Operand
– A (N)	2nd input operand
– R (S)	Operand
– A	Operand
– =	Operand (assignment)

Operand types: F m.n, Q m.n, D m.n ...

The latching function reacts in the following way to changes at the single inputs depending on the function selected:

State at input		Binary output Q
S	R	
0	0	Old state retained
0	1	0
1	0	1
1	1	1 with S/R element 0 with R/S element

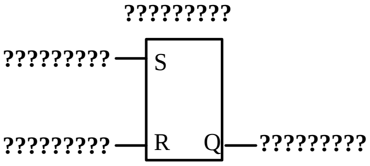
After pressing ***F5*** = *Binary op* and the required function key at the second key level, STEP 5 generates an undefined long box at the position of the long cursor in a LAD statement.

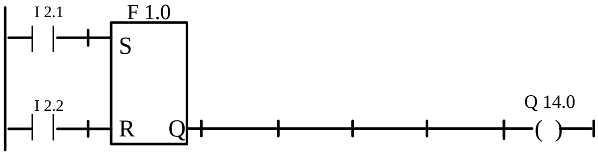
Example

Editing a latching operation with “reset” priority.

1. Position the cursor on an empty element or the contact for the set/reset input and press ***F5*** = *Binary op* and ***F7*** = *S.*

STEP 5 displays the long box or inserts it in the segment.





2. Type in the operand ID for the memory location, in this case ***F 1.0*** and press the ***Return*** key.
3. Enter the input operands with ***F1*** = *NO contacts* ***I 2.1*** and ***I 2.2***. *E*xit each input field with the ***Return*** key.
4. Type in the output (Q) for scanning the binary signal state, in this case ***Q 14.0*** and press the ***Return*** key. Following this, press the ***Insert*** key. Alternatively, press ***F4*** = *–()–*, and then type in ***Q 14.0*** and press the ***Return*** key.

Conversion Operations

(6) in Table 3-3
Conversion operations (BINARY ↔ BCD, 1’s/2’s complement) belong to the supplementary operations (only FB, FX).
A conversion operation is displayed in the empty segment as a long box with the operand in ACCU 1 to the left before the conversion and the result to the right. After pressing ***SHIFT F4*** = *Convert* and selecting the required function at the second key level, STEP 5 generates the long box in which you can enter the operation.

This function corresponds to the statements (STL):

- load operand
- convert the operand
- transfer the result to the operand (ACCU 1)

Operand types: DW, DD, IW...

After generating the long box (see above) you must simply type over the token operands [?????].

Comparator Operations

(7) in Table 3-3
The comparator operations combine two digital operands in ACCU 1 and ACCU 2 to produce a binary result in ACCU 1. The function corresponds to the statements (STL):

- load operand 1
- load operand 2
- execute the selected comparison
- result of logic operation.

A comparison is represented in an empty segment as a long box with the operands in ACCU 1 and 2 to the left and the result of the comparison to the right.

After pressing ***SHIFT F5*** = *Compare* and selecting the required function at the second key level, STEP 5 generates the undefined long box in which you can enter the selected operation.

The selected comparator operation (! =, ><, >=, >, <=, <) is entered in the left-hand side of the long box and the format of the operands to the right, as follows:

F = fixed point number (16 bits)
D = double word (32 bits)
G = floating point number (32 bits)

Note

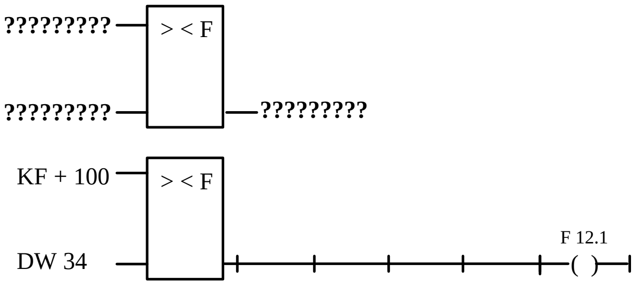
You can only exit a segment or block when all the names and paramenters have been input correctly.

Example

Operation to compare two fixed point numbers:

1. Open a new segment with ********* or ***F6*** *= Compl seg* and then press ***SHIFT F5*** *= Compare.*
2. Select the required operation, in this case ***F2*** *= >< Compare* for *"not equal to"*.

STEP 5 displays the long box with token inputs/outputs and the default operand format "F".



3. Confirm the operand format with the ***Return*** key.
4. Type in the first operand, in this case ***KF + 100***, and press the ***Return*** key.
5. Type in the second operand, in this case ***DW 34***, and press the ***Return*** key.
6. With the cursor on the output, press ***F4*** *= –()–*.
7. Identify the operand to which the result will be assigned, in this case ***F 12.1***, and press the ***Return*** key.

The segment then appears as shown above.

Digital Logic Operations

(8) in Table 3-3
Digital logic operations belong to the supplementary operations (only FB, FX). The operators AND, OR and XOR combine two digital operands in ACCU 1 and ACCU 2 and the result is entered in ACCU 1.
The functions correspond to the statements:

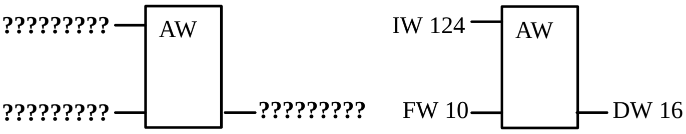
- load operand 1 (DW, IW, FW...),
- load operand 2 (DW, IW, FW...),
- combine the operands as words (AW, OW, XOW),
- transfer the result to operand (DW, IW, FW...).

Example

AND operation of two operands in words.

1. Open a segment with ********* or ***F6*** = *Compl seg* and then press ***SHIFT F1*** = *Math.*
2. Select the required function, here ***F5*** = *AND.*

STEP 5 displays the long box with the token inputs and outputs and the selected format “AW”.



3. Confirm the operand format with the ***Return*** key.
4. Type in the first operation, in this case ***IW 124***, and press the ***Return*** key.
5. Type in the second operand, in this case ***FW 10*** and press the ***Return*** key.
6. Identify the operand to which the result will be transferred, in this case ***DW 16*** and press the ***Return*** key.

The segment then appears as shown on the right-hand side of the figure.

Counter Operations

(9) in Table 3-3
A counter operation is displayed as a long box in the empty segment. The counter operand is above the box. Depending on your selection at the second key level, ***F1*** = “count down”, ***F2*** = “count up”, the first input of the counter input is either a decrementing counter CD or an incrementing counter CU and the second input is the opposite of the first. This results from the rule that the first input of a counter must always be “connected”.

After pressing ***F5*** = *Binary op* and selecting the required function at the second key level, STEP 5 generates the “undefined” long box with the following inputs/outputs:

CD	Decrement the counter value by one when the RLO changes from 0 to 1 at this input (positive going edge).
CU	Increment the counter value by one when the RLO changes from 0 to 1 at this input.
S	Load the counter value from input CV when there is a positive signal change (0 → 1) at the “set” input S.
CV	Value to which the counter is set, decimal (BCD) coded 0 ... 999, operand type: KC, IW, FW, QW, DW.
R	Reset the counter to the value 0 when there is a 1 at this input. The output Q is set to “0”.
BI	Current counter value in binary.
DE	Current counter value in BCD.
Q	The output indicates whether the counter value is zero = “0” or > zero: = “1”.

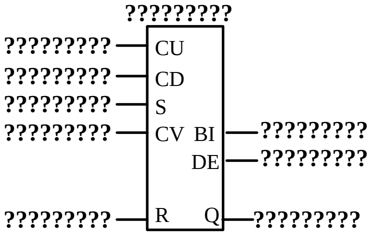
Counter operand:	C 0 ... C 255
Range of values:	0 ... 999

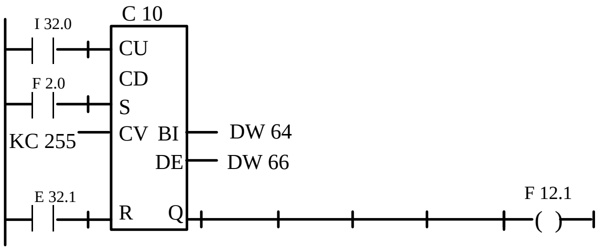
Example

Editing a counter function “count up”.

1. *Open a segment with* ********* *or* ***F6*** *= Compl seg and then press* ***F5*** *= Binary op and* ***F2*** *= CU.*

STEP 5 displays the long box with the token inputs/outputs.





2. Type in the operand (C10) and press the ***Return*** key.
3. Type in the operation for CU, in this case press ***F1*** *= NO contact* and type in ***I 32.0***. Complete the input with the ***Return*** key.
4. Skip the operation for CD by pressing the ***DEL*** key.
5. Type in the operation for setting the counter, in this case press ***F1*** *= NO contact*, and type in ***F 2.0***. Complete the input by pressing the ***Return*** key.
6. Type in the counter value, in this case ***KC 255***, and press the ***Return*** key.
7. Press ***F1*** *= NO contact* to reset input and type in the operand identifier ***I 32.1***, and press the ***Return*** key.
8. Type in the transfer of the counter value to the operands ***DW 64*** and ***DW 66*** and press the ***Return*** key.
9. Press ***F4*** *= –()–* at output Q and type in ***F 12.1*** and press the ***Return*** key.

Timer Operations

(10) in Table 3-3
Using the timer operations, you can program timed program sequences and monitoring functions. You select the required timer function by pressing ***F5*** and selecting the function at the second key level with ***SHIFT F1*** ... ***SHIFT F5***. STEP 5 enters the selected function in symbolic form at the start input of the long box. The timer operand is above the box.

A timer function is started when the RLO at the start input changes. With an OFF delay (SF) the RLO must change from 1 to 0, in all other cases from 0 to 1. The parameters at the start input have the following meaning (see also ***SHIFT F8*** = *Help*):

Symbol	Key	Meaning
1 – - –	***SHIFT F1*** *(SP)*	Start timer as pulse
1 – - V	***SHIFT F2*** *(SE)*	Start timer as extended pulse
T ! – !0	***SHIFT F3*** *(SD)*	Start timer as ON delay
T ! – !S	***SHIFT F4*** *(SS)*	Start timer as stored ON delay
0 ! – !T	***SHIFT F5*** *(SF)*	Start timer as OFF delay

After pressing ***F5*** = *Binary op* and selecting the required function at the second key level, STEP 5 generates the undefined long box with the following inputs/outputs:

”Symbol” Operand for starting the timer function (the symbol corresponding to the timer function is shown in the table above).

TV Input for inputting the timer value.
Operand type: KT, IW, DW ...
The time is a combination of the timer value and the time base. The timer value represents the number of time periods for which the timer function is active. The time base specifies the interval at which the timer value is changed.
e.g. KT = n.i;
n = timer value: 0 ... 999;
i = time base: 0 = 0.01s, 1 = 0.1s, 2 = 1s, 3 = 10s.

R Reset input for the timer function. When this operand changes to 1, the timer and Q are set to 0.

BI Current timer value, binary coded.

DE Current timer value, BCD coded.

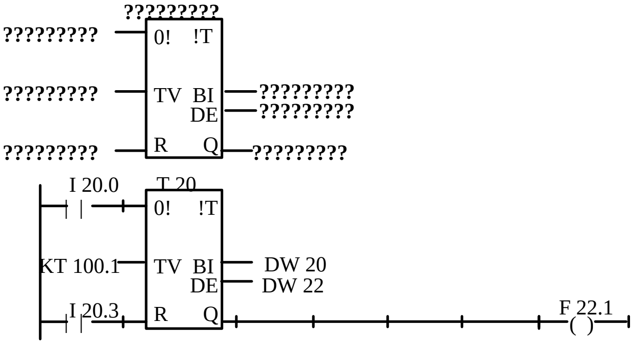
Q Output indicating that the timer is running (Q = 1) or stopped or elapsed (Q = 0).
Timer number: T 0 ... T 255

Example

Editing a timer function with OFF delay.

1. Open a segment with ****** or F6** = Compl seg* and then press ***F5** = Binary op+ **SHIFT F5** = SF.*

STEP 5 displays the long box



2. Type in the timer number, in this case ***T 20*** and press the ***Return*** key.
3. Press ***F1** = NO contact* as the operand to start the timer and type in ***I 20.0*** and press the ***Return*** key.
4. *T*ype in the time ***KT 100.1*** (10s) and press the ***Return*** key.
5. Press ***F1** = NO contact*, type in the reset input ***I 20.3***, and press the ***Return*** key.
6. Enter the transfer of the timer value to the operands ***DW 20*** and ***DW 22*** and complete each input with the Return key.
7. Press ***F4** = –()–* at output Q, type in ***F 22.1*** and press the ***Return*** key.

3.3.4 Editing Control System Flowcharts

Editor

STEP 5 block

in the program file
in the PLC

In the Control System Flowchart method of representation (CSF) the control task is described by connecting function symbols. Based on the circuit logic symbols complying with DIN 40700, the block functions are displayed on the screen with operation symbols (DIN 40719, DIN 19339).

You can program in the Control System Flowchart representation in the following STEP 5 blocks:

- organization block OB
- program block PB
- sequence block SB
- function block FB
- extended function block FX.

STEP 5 stores the corresponding segment comments in the blocks OBDO.nnn, PBDO.nnn etc. Segment titles are stored in the comment blocks OC, PC etc.

It is advisable to enter and correct comments when editing a block and not to write them directly in the documentation or comment blocks.

New Functions

Compared with previous STEP 5 versions, the following functions are now available when editing in CSF:

1. The function key menu has been extended from 8 to 16 function keys and can be activated with the mouse.
2. Function blocks (FB, FX) can be created, however, with the following restrictions :
   - the first segment must be edited in STL
   - no formal operands must be used
3. The following extended functions can be called with softkeys:
   - conditional and unconditional block calls for OB, PB, SB, DB, DX, FB, FX
   - loading and transfer operations
   - negated connector
   - shift and rotate operations (only in FB, FX)
   - conversion operations (only in FB, FX)
   - digital logic operations (only in FB, FX)

Working with the CSF Editor

Settings for editing → ***Object, Settings, Page 1 and 2*** *then*

Program file:	Name of the currently selected program file.
Symbols file:	If you want to work with symbolic operands, you must specify the symbols file, select "symbols: yes" on page 2 and specify the symbol length.
Mode:	"Online" if a PLC is connected and you want to edit in the PLC.
Representation:	"CSF"
Comments:	"Yes" and the comment length (max. 40 characters) if you want to edit segment titles and comments.

If you are editing existing blocks, you can select CSF with ***SHIFT F5*** *= select CSF* regardless of the set method of representation.

Calling the Editor

After selecting the editor function "Edit STEP 5 block(s) in the program file" or "in the PLC" the job box is displayed. Here, you name the block and it is advisable to select the options "Update XRF" and "Update seq. source file" if you are working with symbolic operands.
After confirming your entries with ***OK***, the Control System Flowchart editor is called. A working field appears on the screen (*Figure* 3-21) and the function key menu with the symbols for entering contacts and processing CSF segments.

The editing field is divided into lines and columns in which you enter CSF symbols using the function keys menu or the mouse. A symbol itself takes up one column width. The identifiers of the inputs and outputs before and after the symbol take up a further column. As you build up your segment, you are supported intensively by STEP 5. Connections and symbols of all types (e.g. symbol inputs/outputs for counter or arithmetic functions) are generated automatically and can be reached with the automatic cursor control. STEP 5 rejects inconsistent configurations.

Screen Layout

The screen is divided into 48 fields (8 columns and 6 horizontal sections). The horizontal sections are 3 lines high. CSF symbols are edited in the columns 2 to 7.

The Control System Flowchart can be a maximum of 2.5 times the length of one screen. You can press ***SHIFT F8*** = *Help* to obtain an explanation of the function keys on the screen.

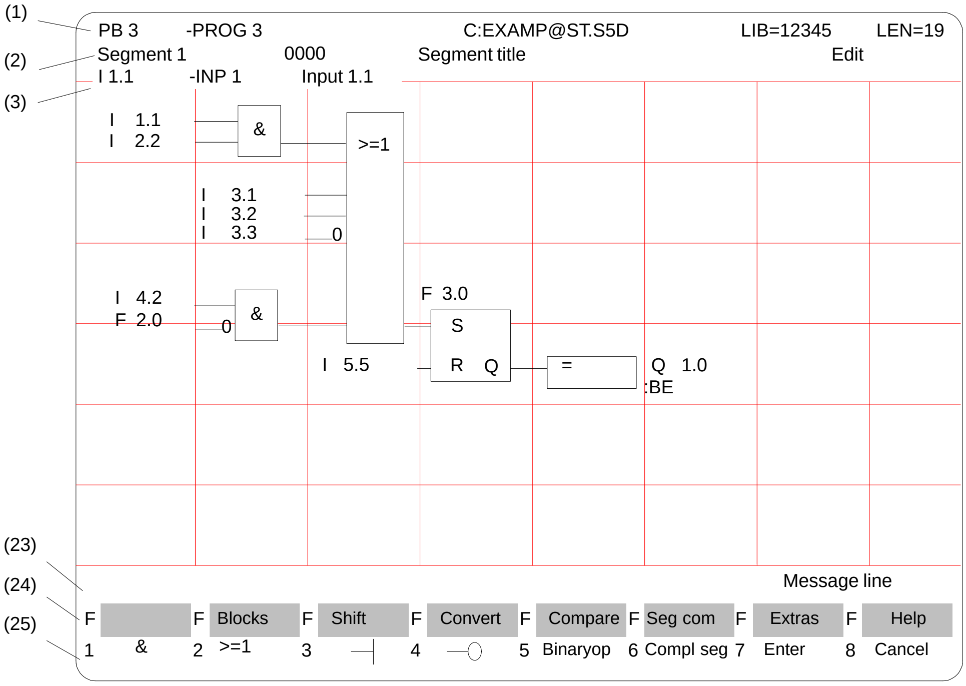

(1)
(2)
(3)
(23)
(24)
(25)
PB 3 -PROG 3 C:EXAMP@ST.S5D LIB=12345 LEN=19
Segment 1 0000 Segment title Edit
I 1.1 -INP 1 Input 1.1
I 1.1
I 2.2
&
>=1
I 3.1
I 3.2
I 3.3
0
I 4.2
F 2.0
0
&
F 3.0
S
R Q
I 5.5
=
Q 1.0
:BE
Message line
F Blocks F Shift F Convert F Compare F Seg com F Extras F Help
1 & 2 >=1 3 4 5 Binaryop 6 Compl seg 7 Enter 8 Cancel

Figure 3-21 Segment in Control System Flowchart (Example)

Table 3-4 Explanation of the Screen Lines (*Figure* 3-21)

Line	Display	Explanation
(1)	PB3 –PROG3 C:EXAMP@ST.S5D LIB=12345 LEN=19	Block type and number Symbolic block name Drive and program file Library number Block length in words
(2)	Segment 1 Segment title Edit	Segment number Text with max. 32 characters STEP 5 mode
(3)	Symbolic operands	Assignment “absolute operand” → “symb. operand” → operand comment, when the cursor is located on an operand identifier
(4)... (22)	Editing area	Input fields for logic operations, calls and operands
(23)	Message line	STEP 5 messages or prompts (red or on a black background)
(24) (25)	Function keys	Key assignment for the currently active functions

Naming Operands

After you input a CSF symbol, the cursor jumps to the name field (max. 8 characters) for the operand. If you have selected a symbol length greater than 8 characters in the “Settings”, STEP 5 only displays the first 8 characters. If you use longer symbolic operand names, make sure that they are unique within the first 8 characters.

Example: you have the following assignment:

Operand	Symbol	Comment
F 100.1	Myflag 100	
F 1.1	Myflag 1.1	
F 1.7	Myflag 1.7	

In CSF, the selected symbolic operand names are all displayed or printed as “Myflag1”.

There are two methods of naming operands, as follows:

1. The operand can be named immediately after selecting a symbol (automatic cursor positioning),
   or if you have exited the name field [?????], you can return to it with the ***Return*** key.
2. Entering the operand names in the name fields of the completed segment, guided by the long cursor.

Note

You can only exit a segment or block when all the names and parameters have been input correctly (make sure the formats are correct).

Reconfiguring a Segment

If, while editing a segment its layout has become awkward (e.g. as a result of repeated branches) you can redisplay the segment by pressing the ***SHIFT F7*** = *Extras* and ***F2*** = *New disp* even if the segment does not yet have all the required parameters. The screen is then reconfigured and the display layout optimized.

Editing Symbolic Operands

When you press ***F1*** = *Disp symb* in the output mode, STEP 5 displays a list of operands in absolute and symbolic form for the open segment. You can then edit this list. If you use longer symbol names, make sure that the names are unique within the first 8 characters. The symbolic operand names are reduced to 8 characters on the screen and when printing in LAD and CSF.

If you make changes, it is advisable to update the sequential source file.

Logic Operations

When you select “CSF” on page 2 of the “Settings” and “Edit STEP 5 block(s) in ...”, STEP 5 opens the block selected in the job box at segment 1. If you are working with a new block, this is empty.

Using the function keys, you can now input the basic CSF symbols for AND/OR operations on binary operands (Table 3-5). The left-hand column of this table contains the operation for processing the operands which you call in the edit mode using the keystrokes shown.

Table 3-5 Logic operations in CSF (Control System Flowchart)

Operation	Function keys	Explanation
&	**F1**	AND operation
> = 1	**F2**	OR operation
——┤	**F3**	Input
——○	**F4**	Negated input
Binary op	**F5**	Call complex functions
#	**F5 and F4**	Connector
/	**F5 and F5**	Negated, connector

Editing, Modifying and Deleting Functions

When you input the first operator at the position marked by the long cursor in the empty segment, a function block is created with two input operands and one output. You can create a serial chain of functions with a maximum of 5 AND/OR operators.

Modifying a Segment

The number of input operands can be increased (see example):

- You can append by positioning the long cursor below the lowest input of the long box.
- You can insert and position in a function block. (limit = 2 1/2 times the screen height)

You can convert an input to a function block:

- Place the cursor on the corresponding operand identifier and press ***F1*** = & or ***F2*** = >=1.

You invert an input by positioning the cursor on the operand identifier and pressing ***F4*** *= Input* or ***F5*** *= Negated input*. The current input then has the opposite effect to the previous one. You can modify an edited function by positioning the cursor on the function identifier in the box and overwriting it with the required operation.

Deleting

The following rules apply when deleting operands and functions in segments (***DEL***):

1. An input located under the long cursor is deleted. The function block itself is reduced in length by one line (A).
2. If you delete a connected input, the function element before this input is also removed. The input is then displayed as non-connected (B).
3. A function element with two operand inputs is removed. The remaining operand then occupies the free input of the next block (C).
4. Function elements with two inputs (one of which is connected) are removed from the segment after deleting the operand. The function elements before the other input now influence the next block directly.

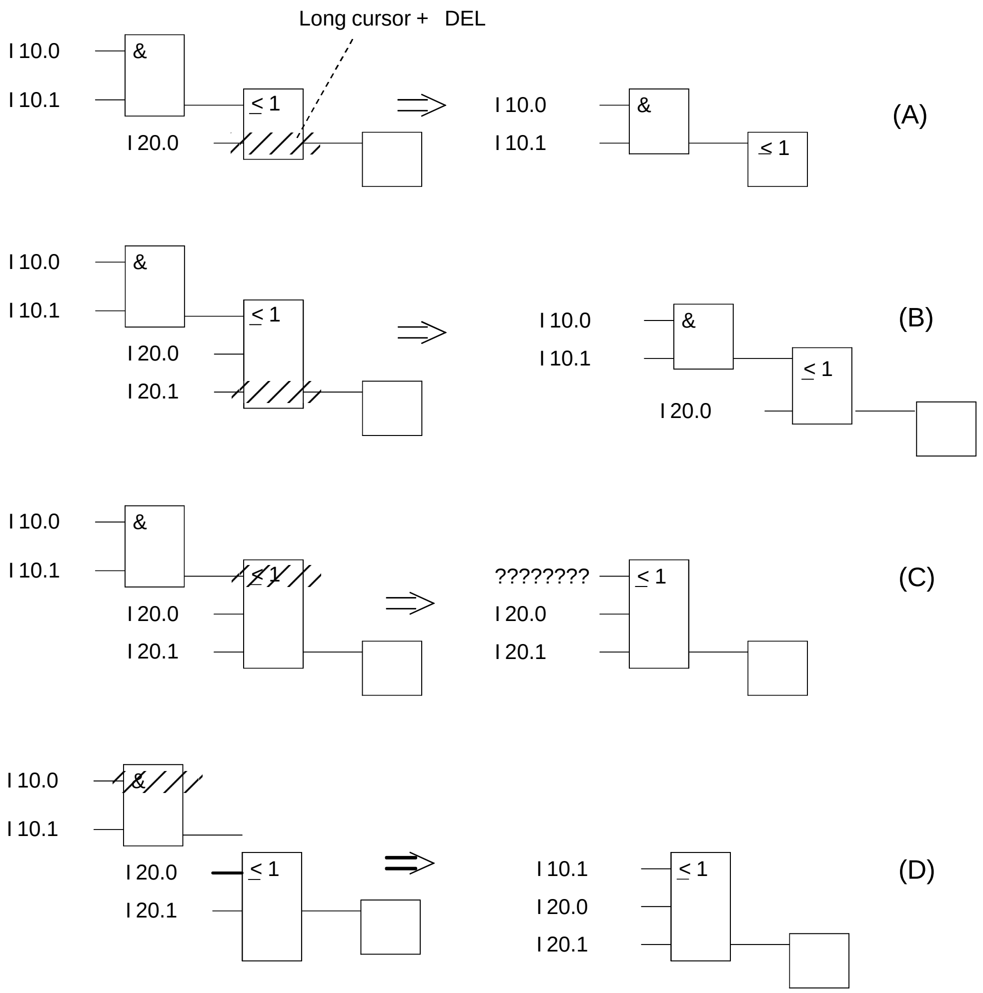


Figure 3-22 Deleting Operands and Functions (Example)

If you want to mark a named input operand as undefined, it is sufficient to type in a question mark as the first character of the input field.

Appending Operands

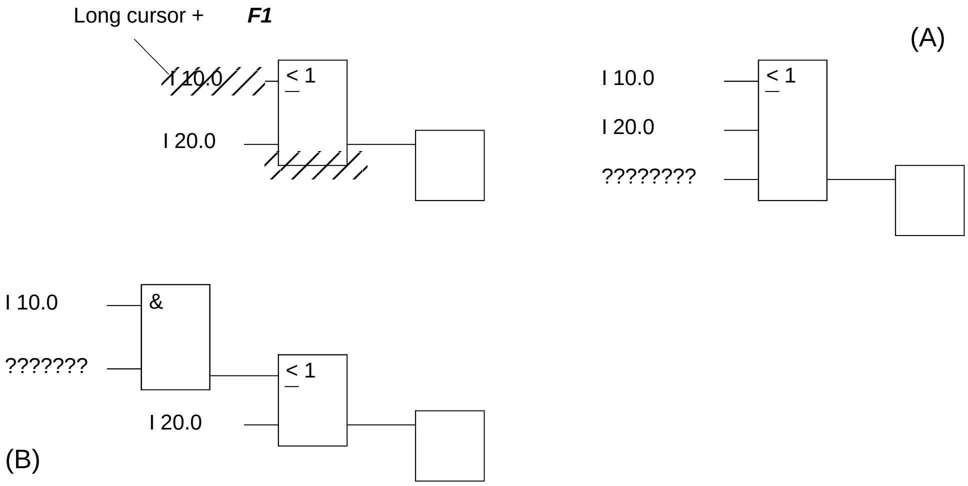
Position the long cursor on the lower edge of the function block and press ***F3***.

An undefined operand is appended to the bottom of the block (A).

Appending a Function Block

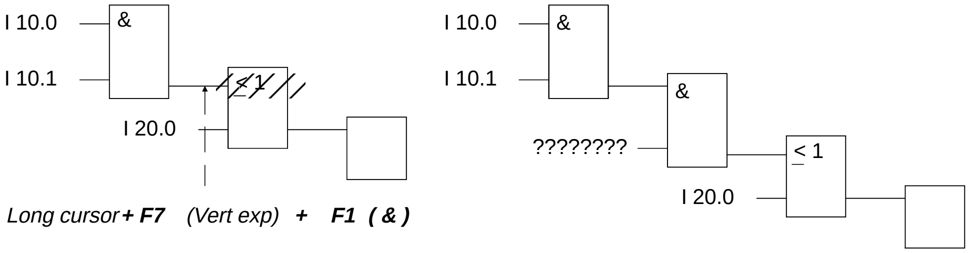
Position the long cursor on the input operand to be replaced by a function block and press ***F1*** or ***F2***.

STEP 5 places the selected function block with two inputs (if necessary with implicit expanding) before the previous input. The operand identifier is transferred to the upper input of the new block (B).
Horizontal and vertical expanding, i.e. in this case moving the segment to the right and down is performed implicitly.



Inserting Operands

1. Position the long cursor on the input of the function block above which you want to insert an input operand.
2. Press ***SHIFT F7*** *= Extras,* ***F7*** *= Vert exp* and then ***F3*** *= Input.*

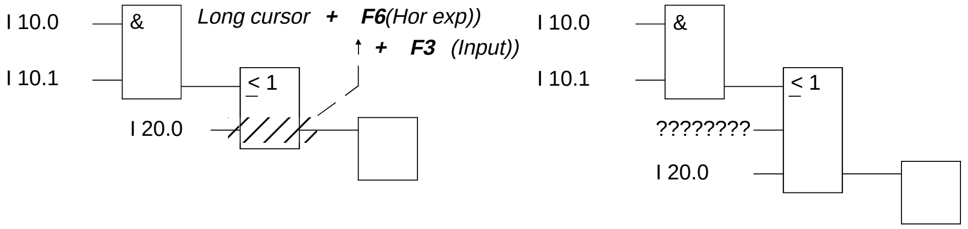


Long cursor ***+ F7*** *(Vert exp)* ***+ F1 (&)***

A non-connected operand is inserted in the block. After you have named the operand, you can invert the input with ***F4***.

Inserting a Function Block

1. Position the long cursor on the input of the box before which you want to insert a new function.
2. Press ***SHIFT F7*** *= Extras,* ***F6*** *= Hor exp* and select the required function, in this case ***F1*** *= &.*



Long cursor ***+ F6****(Hor exp))*
↑ ***+ F3*** *(Input))*

STEP 5 places the selected function block so that the upper input is connected. The operand at the lower input is undefined.

Editing Connectors

Connectors and negated connectors (Table 3-2) are intermediate flags in binary logic operations. A connector is input in CSF like a function block. If it follows the last block of a segment it is handled and displayed as an output.

Inserting

You want the intermediate result written to a flag F20.1 at the output of the AND block.

1. Position the cursor on the input of the next block and press ***SHIFT F7*** *= Extras,* ***F6*** *= Hor exp and* ***F5*** *= Binary op* and ***F4*** *= # .*
2. Name the connector, e.g. ***F 20.1*** *(A)* and press the ***Return*** key.

Connector Stack

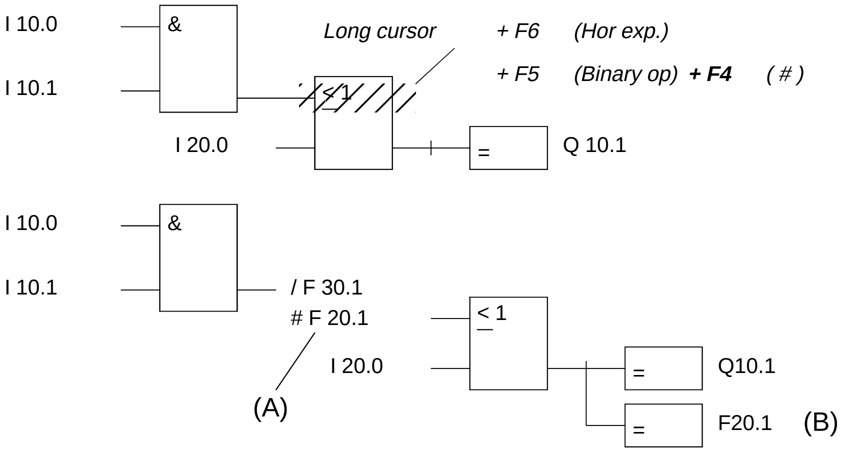
You obtain a connector stack by

Positioning the cursor on the connector and pressing **F5 =** Binary op again and ***F4*** *= # or* ***F5*** = / and typing in the flag name, in this case ***F 30.1***.

With implicit expanding, the previously entered connector is moved one line down.

Connector before Output

Inputting connector F 20.1 before the output results in the situation shown in (B).



You can delete a connector by positioning the cursor on the connector and pressing ***DEL***.

Complex Functions

In CSF, the following rules apply to the complex functions listed in Table 3-6:

1. All operations (1) to (10) in Table 3-6 are represented as “long boxes” in which the operands are displayed on the left before the processing and on the right the result of the processing. STEP 5 enters the operation selected with the function keys in the long box itself.
2. Combinations of several complex functions are possible in a segment. Make sure, however, that the data types match up. A combination of complex function elements with binary function elements is only possible with the complex element “comparator”. Parallel branches are not allowed.
3. Some function elements can be extended, i.e. the number of inputs can be increased provided the operation allows.
4. The “shift/rotate” function (4) requires the shift parameter “n” to be entered in the long box, i.e. the number of bits by which the content of the operand is shifted left or right. The maximum possible shift depends on the format of the operand (16 or 32 bits).
5. With the functions “Math” and “Compare” you can specify a different operand type in the long box. The type “fixed point number = F” is the default.

Note

The type can only be changed once directly after calling the long box.

In the editing mode, the following functions can be called with ***SHIFT*** and function keys or ***F5***:

(1) Arithmetic operations
(2) Block calls
(3) Load and transfer operations
(4) Shift and rotate word/double word operands
(5) Binary operations
(6) Conversion operations
(7) Comparison between two operands
(8) Digital logic operations
(9/10) Counter and timer operations

Table 3-6 Complex Functions in CSF

Operation	**Keys (function keys)**		**Explanation**
Math. ADD, SUB MULT, DIV	**SHIFT F1 and**	***F1, F2*** ***F3, F4***	(1) **Arithmetic operations:** Addition, subtraction multiplication, division
(with FBs/FXs) AND OR XOR	**SHIFT F1** **and**	**F5** **F6** **F7**	*(8)* **Digital logic operations*:*** AND operation, words OR operation, words Exclusive OR operation, words
Blocks JU FB, JC FB DO FX, DOC FX JU..., JC... C DB, CX DX	**SHIFT F2** **and**	**F1, SHIFT *F1 F2,*** **SHIFT *F2 F4,*** **SHIFT *F4 F6,*** **SHIFT *F6***	(2) **Call blocks as follows:** FB unconditional, FB conditional FX unconditional, FX conditional OB, PB, SB unconditional, conditional DB, DX
(Shift) L/T	**SHIFT F3** **and**	**F7**	*(3)* **Load and transfer operations** Load and transfer operand
SHIFT (with FBs/FXs) SLW, SLD SRW SSW, SSD RLD, RRD	**SHIFT F3** **and**	**F1, SHIFT F1 F2** **F3, SHIFT F3** **SHIFT F4, SHIFT F5**	(4) **SHIFT and rotate operations** SHIFT word/double word left SHIFT word right SHIFT word/double word with sign right Rotate left, right
Convert (with FBs/FXs) DEF, CFW DUF, CSW DED, CSD DUD FDG, GFD	**SHIFT F4** **and**	**F1, SHIFT F1 F2,** **SHIFT F2 F3,** **SHIFT F3 F4** **F5, F6**	(6) **Convert operations** BCD->binary, form 1’s compl., 16 bit Binary->BCD, form 2’s comp., 16 bit BCD->binary, form 2’s compl., 32 bit Binary->BCD, 32 bit Fixp -> floatp, floatp -> fixp, 32 bit

Table 3-6 Complex Functions in CSF

Operation	Keys (function keys)		Explanation
Compare ! = > < > = < = > <	**SHIFT F5** **and**	**F1, F2 F3,** **F5 F4, F6**	*(7)* **Comparator operations (between two operands):** Compare for “equal to”, **“**not equal to” Compare for greater than or equal to, less than or equal to Compare for “greater than”, “less than”
Binary op CD, CU	**F5** **and**	**F1, F2**	(9) **Counter operations:** Counter value incremented, decremented by 1
Binary op SP, SE SD, SF SS	**F5** **and**	**SHIFT F1/F2** **SHIFT F3/F5** **SHIFT F4**	(10) **Timer operations:** Start timer as pulse, extended pulse Start timer as ON/OFF delay Start timer as stored on delay
R/S S/R	**F5 and**	**F6** **F7**	*(5)* **Binary latching operations:** Priority resetting flip-flop Priority setting flip-flop

Arithmetic Operations

(1) in Table 3-6
The operators ADD, SUB, MULT and DIV combine two operands in ACCU 1 and 2 to produce a result in ACCU 1.

Arithmetic operations can be cascaded with other complex functions.
At the highest input:

- arithmetic operations
- shift operations
- conversion operations
- digital logic operations

At the output:

- arithmetic operations
- shift operations
- conversion operations
- comparator operations
- digital logic operations

The arithmetic operations correspond to the statements (STL):

- load operand 1;
- load operand 2;
- execute the required logic operation;
- transfer result to operand (ACCU 1).

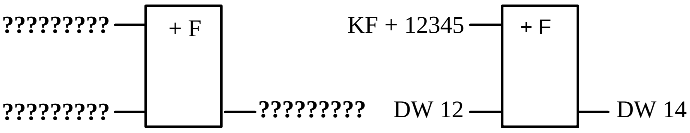
Operand types : KF, DW, IW...

Examples

Editing an ADD operation for two fixed point numbers:

1. Press ******* or ***F6*** = *Compl seg* and then ***SHIFT F1*** = *Math.*
2. Select the required operation, here ***F1*** = *ADD.*

STEP 5 displays the long box with undefined inputs and outputs and the default operand format “F”.

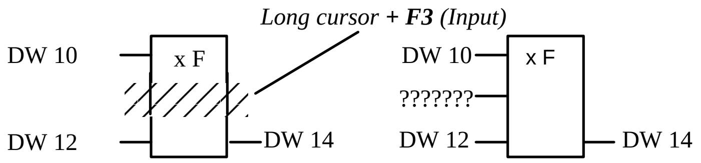


3. Confirm the operand format with the ***Return*** key.
4. Type in the 1st operand, in this case KF + 12345 and press the ***Return*** key.
5. Type in the 2nd operand, in this case DW 12 and press the ***Return*** key.
6. Name the operand to which the result will be transferred (DW 14) and press the ***Return*** key.

The segment now appears as shown on the right-hand side of the figure.

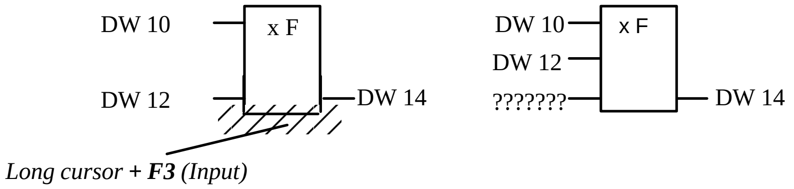
Inserting an Input

*Position the long cursor between the two inputs and press **F3** = Input and name the input.*


Long cursor **+ *F3*** *(Input)*

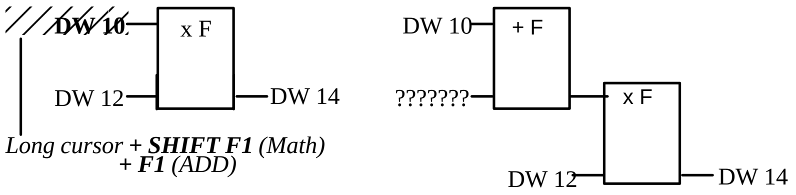
Appending an Input

*Position the long cursor at the bottom of the function box and press **F3** = Input and name the input.*


Long cursor **+ *F3*** *(Input)*

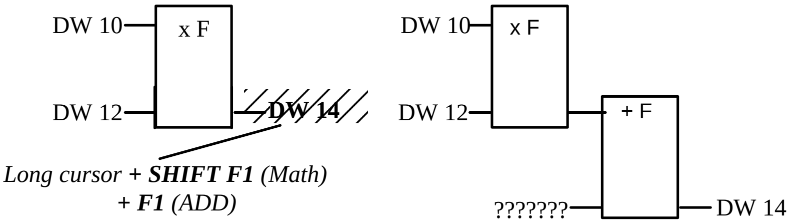
Inserting a complex function at the input

*Position the long cursor on the 1st input operand, select a complex function, in this case **SHIFT F1** = Math and **F1** = ADD and label the operand*


Long cursor **+ *SHIFT F1*** *(Math)*
+ *F1* *(ADD)*

Inserting a Complex Function at the Output

Position the long cursor on the output operand, select a complex function, in this case ***SHIFT F1*** *= Math and* ***F1*** *= ADD and label the operand*



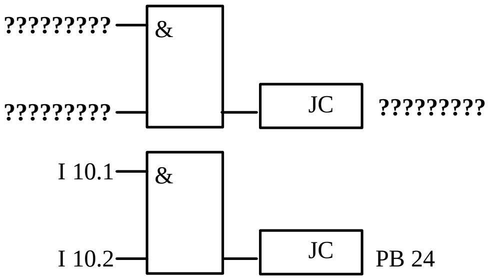
Block Calls

(2) in Table 3-6
Using block calls in STEP 5, you can call further blocks in the user program from any block allowing a structured program sequence. A block call is programmed in CSF as a long box. Only one block call per segment is allowed.
In an empty segment, you can enter a block call directly using the softkeys.

Example 1

Conditional program block call.

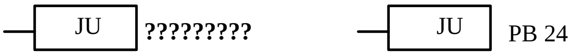
1. Press ***SHIFT F2*** *= Blocks* and ***SHIFT F4*** *= JC..* in the empty segment.
2. Type in the input operands, in this case ***I 10.1*** and ***I 10.2.*** Specify the destination block, in this case ***PB 24*** in the right input field and complete with the ***Return*** key.



Example 2

Unconditional program block call

1. Press ***SHIFT F2*** = *Blocks* and ***F4*** = *JU..* in the empty segment.
2. Specify the destination block, in this case ***PB 24,*** in the right input field and complete with the ***Return*** key.



Example 3

Unconditional FB call in an empty segment

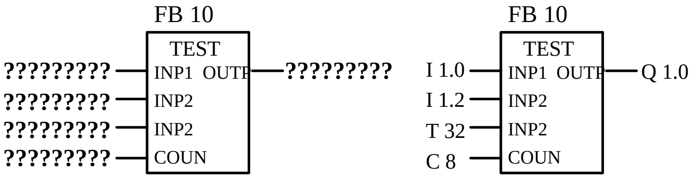
1. Press ***SHIFT F2*** = *Blocks* and ***F1*** = *JU FB.*

The editor displays the “roof” of the block with the cursor in the labelling field.

2. Type in the name of the function block to be called, in this case ***FB 10***.

The function block with its formal operands is displayed.

3. Type in the name in absolute or symbolic form. Move to the other fields using the ***Return*** key.



The segment then appears as shown on the right-hand side.

Loading and Transfer Operations

(3) in Table 3-6
The function is displayed as a “long box” with the operand to the left and the result to the right. The function ***SHIFT F3*** = *Shift* and ***F7*** = *L/T* correspond to the following STL statements:

- load operand (DW, DD, IW...),
- transfer to operand (DW, DD, IW...).

After generating the long box (see above) you simply enter the operands displayed as [?????].

Shift and Rotate Operations

(4) in Table 3-6
Shift and rotate operations belong to the supplementary operations (only FB, FX). A shift/rotate operation is displayed in an empty segment as a long box with the operand in ACCU 1 to the left before the shift operation and the result to the right.
They can be cascaded with other complex functions at the input and output.
After pressing the softkeys ***SHIFT F3*** = *Shift* and the required function at the second key level, STEP 5 generates the “undefined” long box in which you enter the required operation.

The character cursor flashes below the parameter “n”. Here, you enter the number of bits by which the content of the operand will be shifted.
The function corresponds to the STL statements:

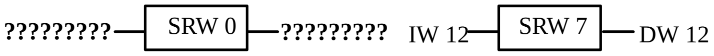
- load operand
- shift/rotate operand by “n” bits
- transfer result to operand (ACCU 1).

Example

Shifting the input operand IW 12 seven bits to the right and transferring to DW 12.

1. Press ****** or F6*** = *Compl seg* followed by ***SHIFT F3*** *=Shift.*
2. Select the required operation, in this case ***F2*** = *SRW.*

   STEP 5 displays the long box (left).



3. Position the cursor on the parameter “n” in the box, in this case 0, and type in the number “7”.
4. Type in the input and output operands.

Note

It is not possible to change the parameter “n” later.

Latching Operations

(5) in Table 3-6
Using the latching functions, the RLO can be stored statically outside the processor. You can specify how the latching function works after pressing ***F5*** = *Binary op* and then selecting either ***F6*** "priority set" or ***F7*** "priority reset" at the second key level.
STEP 5 enters the operands with priority at the top of the long box.
The latching function is displayed as a box with 2 inputs and 1 output, S is the "set" input, R is the "reset" input and Q is the output. Only one latching function can be inserted in a segment.

The latching function corresponds to the following statements (STL):

– A (N)	1st input operand
– S (R)	Operand
– A	2nd input operand
– R (S)	Operand
– A (N)	Operand
– =	Operand (assignment)

Operand types: F m.n, Q m.n, D m.n ...

The latching function reacts in the following way to changes at the single inputs depending on the function selected:

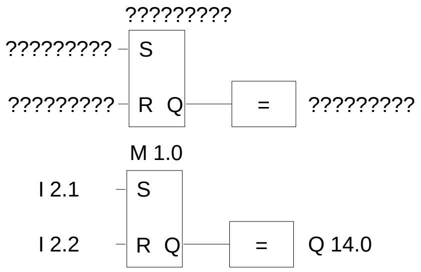
State at input		State at output Q
S	R	
0	0	Old state retained
0	1	0
1	0	1
1	1	1 with S/R flip flop 0 with R/S flip flop

After pressing ***F5*** = *Binary op* and the required function key at the second key level, STEP 5 generates an undefined long box at the position of the long cursor in a CSF segment.

Example

Editing a latching operation with “reset” priority.

1. Press ****** or F6** = Compl seg* and then ***F5** = Binary op* and ***F7** = S.*



2. Type in the operand ID for the memory location, in this case ***F 1.0*** and press the ***Return*** key.
3. If the input operands do not yet exist, type in the NO contacts ***I 2.1*** and ***I 2.2***. Exit each input field with the ***Return*** key.
4. Type in the output for scanning the binary signal state, in this case ***Q 14.0*** and press the ***Return*** key. Following this, press the ***Insert*** key.

Conversion Operations

(6) in Table 3-6
Conversion operations (BINARY ↔ BCD, 1’s/2’s complement) belong to the supplementary operations (only FB, FX). A conversion operation is displayed as a long box with the operand in ACCU 1 to the left before the conversion and the result to the right. They can be cascaded with other complex functions at the input and output.
After pressing ***SHIFT F4** = Convert* and selecting the required function at the second key level, STEP 5 generates the long box in which you can enter the operation.
This function corresponds to the statements (STL):

– load operand
– convert the operand
– transfer the result to the operand (ACCU 1)

Operand types: DW, DD, IW...

After generating the long box (see above) you must simply type over the token operands [?????].

Comparator Operations

(7) in Table 3-6
The comparator operations combine two digital operands in ACCU 1 and ACCU 2 to produce a binary result in ACCU 1. They can be cascaded with other complex functions at the input. The function corresponds to the statements (STL):

- load operand 1
- load operand 2
- execute the selected comparison
- result of logic operation.

A comparison is represented in an empty segment as a long box with the operands in ACCU 1 and 2 to the left and the result of the comparison to the right.

After pressing ***SHIFT F5*** = *Compare* and selecting the required function at the second key level, STEP 5 generates the undefined long box in which you can enter the selected operation.

The selected comparator operation (! =, ><, >=, >, <=, <) is entered in the left-hand side of the long box and the format of the operands to the right, as follows:

F = fixed point number (16 bits)
D = double word (32 bits)
G = floating point number (32 bits)

Note

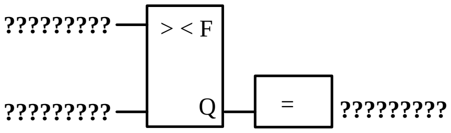
The type can only be changed once directly after calling the long box.

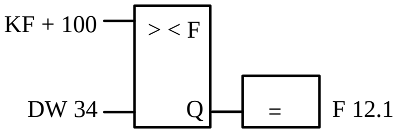
Example

Operation to compare two fixed point numbers:

1. Open a new segment with ******* or ***F6*** = *Compl seg* and then press ***SHIFT F5*** = *Compare.*
2. Select the required operation, in this case ***F2*** = >< compare for “not equal to”.

STEP 5 displays the long box with token inputs/outputs and the default operand format “F”.





3. Confirm the operand format with the ***Return*** key.
4. Type in the first operand, in this case ***KF + 100***, and press the ***Return*** key.
5. Type in the second operand, in this case ***DW 34***, and press the ***Return*** key.
6. Identify the operand to which the result will be assigned, in this case ***F 12.1***, and press the ***Return*** key.

The segment then appears as shown above.

Digital Logic Operations

(8) in Table 3-6
Digital logic operations belong to the supplementary operations (only FB, FX). They can be cascaded with other complex functions such as arithmetic operations.
The operators AND, OR and XOR combine two digital operands in ACCU 1 and ACCU 2 and the result is entered in ACCU 1.
The functions correspond to the statements:

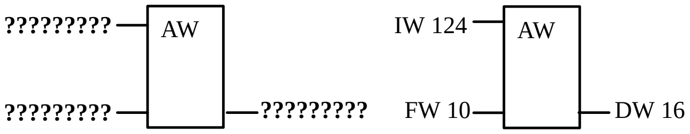
- load operand 1 (DW, IW, FW...),
- load operand 2 (DW, IW, FW...),
- combine the operands as words (AW, OW, XOW),
- transfer the result to operand (DW, IW, FW...).

Example

AND operation of two operands in words.

1. Open a segment with ********* or ***F6*** *= Compl seg* and then press ***SHIFT F1*** *= Math.*
2. Select the required function, here ***F5*** *= AND.*

STEP 5 displays the long box with the token inputs and outputs and the selected format “AW”.



3. Type in the first operation, in this case ***IW 124***, and press the ***Return*** key.
4. Type in the second operand, in this case ***FW 10*** and press the ***Return*** key.
5. Identify the operand to which the result will be transferred, in this case ***DW 16*** and press the ***Return*** key.

The segment then appears as shown on the right-hand side of the figure.

Counter Operations

(9) in Table 3-6
A counter operation is displayed as a long box in the empty segment. The counter operand is above the box. Depending on your selection at the second key level, ***F1*** = “count down”, ***F2*** = “count up”, the first input of the counter input is either a decrementing counter CD or an incrementing counter CU and the second input is the opposite of the first. This results from the rule that the first input of a counter must always be “connected”.

After pressing ***F5*** = *Binary op* and selecting the required function at the second key level, STEP 5 generates the “undefined” long box with the following inputs/outputs:

CD Decrement the counter value by one when the RLO changes from 0 to 1 at this input (positive going edge).

CU Increment the counter value by one when the RLO changes from 0 to 1 at this input.

S Load the counter value from input CV when there is a positive signal change (0 → 1) at the “set” input S.

CV Value to which the counter is set, decimal (BCD) coded 0 ... 999, operand type: KC, IW, FW, QW, DW.

R Reset the counter to the value 0 when there is a 1 at this input. The output Q is set to “0”.

BI Current counter value in binary.

DE Current counter value in BCD.

Q The output indicates whether the counter value is zero = “0” or > zero: = “1”.

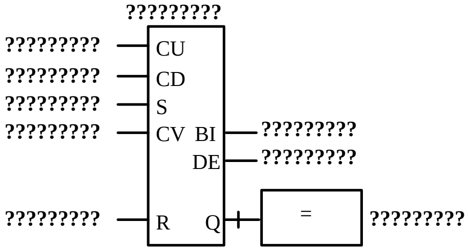
Counter operand:	C 0 ... C 255
Range of values:	0 ... 999

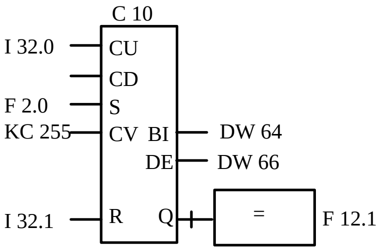
Example

Editing a counter function “count up”.

1. Open a segment with ********* or ***F6*** *= Compl seg* and then press ***F5*** *= Binary op* and ***F2*** *= CU.*

STEP 5 displays the long box with the undefined inputs/outputs.





2. Type in the operand (C10) and press the ***Return*** key.
3. Type in the operand for CU, in this case ***I 32.0***. Complete the input with the ***Return*** key.
4. Skip the operation for CD by pressing the ***DEL*** key.
5. Type in the operand for setting the counter, in this case ***F 2.0***. Complete the input by pressing the ***Return*** key.
6. Type in the counter value, in this case ***KC 255***, and press the ***Return*** key.
7. Type in the reset input, in this case ***I 32.1***, and press the ***Return*** key.
8. Type in the transfer of the counter value to the operands ***DW 64*** and ***DW 66*** and press the ***Return*** key.
9. Type in ***F 12.1*** at the output and press the ***Return*** key.

Timer Operations

(10) in Table 3-6
Using the timer operations, you can program timed program sequences and monitoring functions. You select the required timer function by pressing ***F5*** and selecting the function at the second key level with ***SHIFT F1*** ... ***SHIFT F5***. STEP 5 enters the selected function in symbolic form at the start input of the long box. The timer operand is above the box.

A timer function is started when the RLO at the start input changes. With an OFF delay (SF) the RLO must change from 1 to 0, in all other cases from 0 to 1. The parameters at the start input have the following meaning:

Symbol	**Key**	**Meaning**
1 – - –	***SHIFT F1*** *(SP)*	Start timer as pulse
1 – - V	***SHIFT F2*** *(SE)*	Start timer as extended pulse
1 ! – !0	***SHIFT F3*** *(SD)*	Start timer as ON delay
1 ! – !S	***SHIFT F4*** *(SS)*	Start timer as stored ON delay
0 ! – !T	***SHIFT F5*** *(SF)*	Start timer as OFF delay

After pressing ***F5*** = *Binary op* and selecting the required function at the second key level, STEP 5 generates the undefined long box with the following inputs/outputs:

”Symbol” Operand for starting the timer functions (the symbol corresponding to the timer functions is shown in the table above).

TV Input for inputting the timer value.
Operand type: KT, IW, DW ...
The time is a combination of the timer value and the time base. The timer value represents the number of time periods for which the timer function is active. The time base specifies the interval at which the timer value is changed.
e.g. KT = n.i;
n = timer value: 0 ... 999;
i = time base: 0 = 0.01s, 1 = 0.1s, 2 = 1s, 3 = 10s.

R Reset input for the timer function. When this operand changes to 1, the timer and Q are set to 0.

BI Current timer value, binary coded.

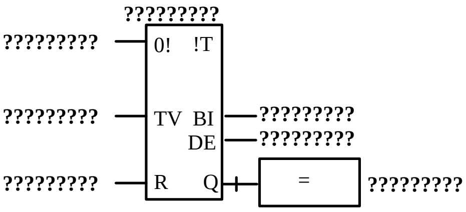
DE Current timer value, BCD coded.

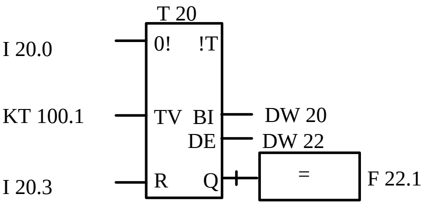
Q Output indicating that the timer is running (Q = 1) or stopped or elapsed (Q = 0).
Timer number: T 0 ... T 255

Example

Editing a timer function with OFF delay.

1. Open a segment with ****** or F6** = Compl seg* and then press ***F5** = Binary op* + ***SHIFT F5** = SF.*





2. Type in the timer number ***T 20*** and press the ***Return*** key.
3. Type in ***I 20.0*** to start the timer and press the ***Return*** key.
4. Type in the time ***KT 100.1** (10s)* and press the ***Return*** key.
5. Type in the reset input ***I 20.3***, and press the ***Return*** key.
6. Enter the transfer of the timer value to the operands ***DW 20*** and ***DW 22*** and complete each input with the ***Return*** key.
7. Type in ***F 22.1*** at the output and press the ***Return*** key.

3.3.5 Editing Data Blocks

Data blocks contain fixed or variable data for the user program to work with.

The block title and line comments are stored in the corresponding comment block DC/DCX. STEP 5 stores a block comment in the documentation block DBDO.nnn/DXDO.nnn.

Both block types are generated automatically when you enter the edited DB/DX. They are not transferred to the PLC or to an EPROM/EEPROM. Although it is possible to edit directly in these blocks, it is advisable to input titles and comment texts in the DB/DX since all the assignments can be recognized here.

After introducing you to the basics of editing a data block, the individual functions of the editor are described separately.

Selecting the Editor

Editor

STEP 5 block

in the program file
in the PLC

(1) The job box is displayed in which you can make your selections (see Figure 3-23). If you want to edit or work on a data block, specify the block either in absolute form (e.g. DB 15) or using a symbolic name.

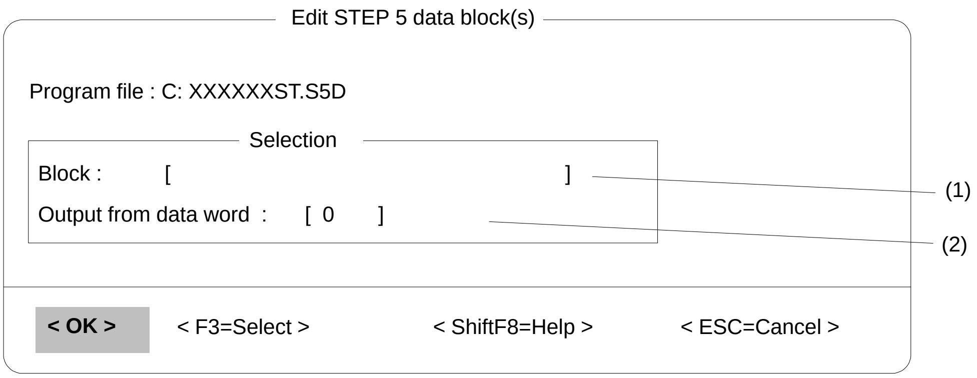
If you want to search for a particular data word in one or more data blocks, enter the block(s) in the block list in absolute form (maximum 6 DBs) or enter one DB with a symbolic name. You can then enter the number of the data word, e.g. 123, in (2).

If you press ***SHIFT F8*** = *Help*, STEP 5 displays a list of possible inputs.
If you want to work with an existing block, you can select this using the → *Block selection box*. You call the block selection box with ***F3*** = *Select*.

Output from Data Word

(2) Here, you can type in a data word number. You exit the field using the ***Return*** key or select a different field with the mouse.

The data word number is searched for in all the blocks you specified.



Edit STEP 5 data block(s)

Program file : C: XXXXXXST.S5D

Selection

Block : [ ]

Output from data word : [0]

< OK > < F3=Select > < ShiftF8=Help > < ESC=Cancel >

Figure 3-23 Editing STEP 5 Data Block(s)

Selecting The Editor and Search Function

Editor

Data block

Search
in the program file
in the PLC

1. The job box is displayed in which you can make your selections (see figure below). If you want to edit or work on a data block, specify the block either in absolute form (e.g. DB 15) or using a symbolic name.

If you want to search for a particular data word in one or more data blocks, enter the block(s) in the block list in absolute form (maximum 6 DBs) or enter one DB with a symbolic name.

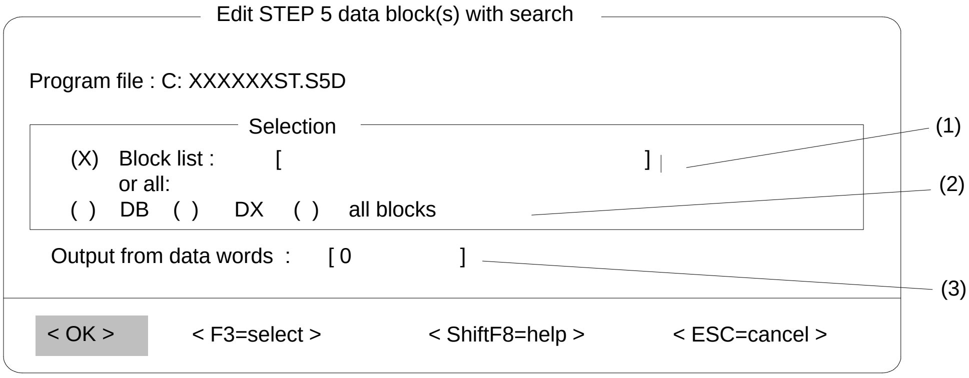
If you press ***SHIFT F8*** = *Help*, STEP 5 displays a list of possible inputs.

If you want to work with an existing block, you can select this using the → *Block selection box*. You call the block selection box with ***F3*** = *Select*.

2. Here, you select a data block type or all data blocks and do not make an entry in the block list.

Output from Data Word

3. Here, you enter the data word number to be searched for. Exit the field with the ***Return*** key or select a different field using the mouse.
The data word number is searched for in all the blocks you have specified.



Edit STEP 5 data block(s) with search

Program file : C: XXXXXXST.S5D

Selection

(X) Block list : [ ]
or all:
( ) DB ( ) DX ( ) all blocks

Output from data words : [0]

Figure 3-24 Editing STEP 5 Data Blocks using the Search Function

The input fields of the DB editor

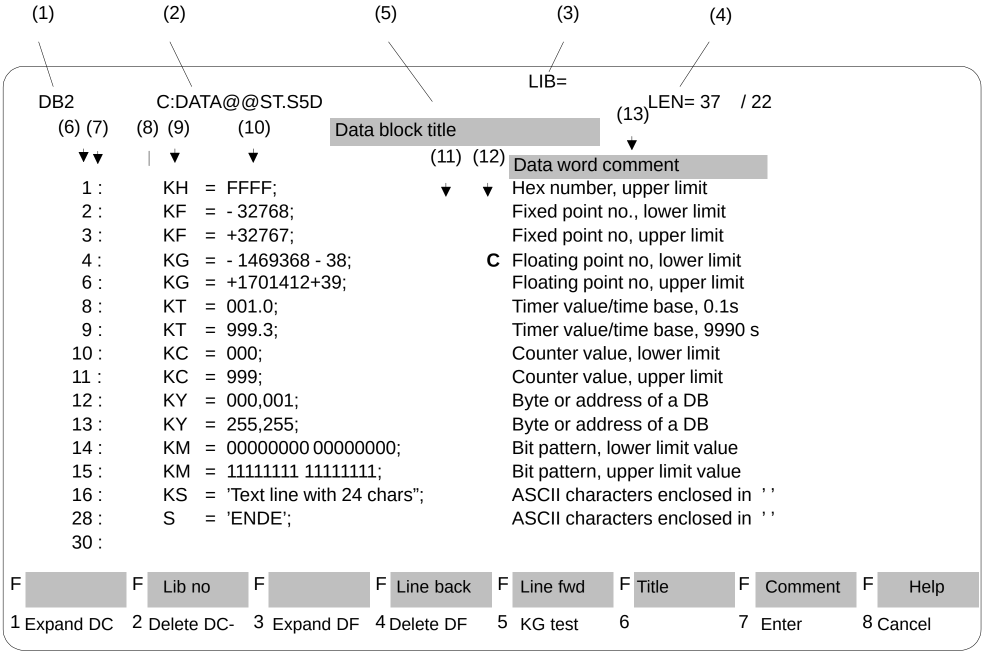
(1) DB field This displays the block number (here: DB 2) that you entered when you filled in the job box.

(2) Program file This field displays the drive and the name of the program file (here: drive C: with the program file DATA@@ST.S5D).

(3) LIB field In this field you can input a maximum 5-digit long DB library number (number from 0 to 99999).

(4) LEN field This field displays the block length in data words, including the block header. The number after the slash is the length of the DB preheader. This display is updated whenever you enter a complete line.

Figure 3-25 shows the editing field of the DB editor with the softkeys of the basic menu and a displayed data block.



```
DB2            C:DATA@@ST.S5D                                  LIB=
                                                                     LEN= 37   / 22
                                     Data block title
                                                              Data word comment
   1 :   KH  =  FFFF;                                         Hex number, upper limit
   2 :   KF  =  - 32768;                                      Fixed point no., lower limit
   3 :   KF  =  +32767;                                       Fixed point no, upper limit
   4 :   KG  =  - 1469368 - 38;                             C Floating point no, lower limit
   6 :   KG  =  +1701412+39;                                  Floating point no, upper limit
   8 :   KT  =  001.0;                                        Timer value/time base, 0.1s
   9 :   KT  =  999.3;                                        Timer value/time base, 9990 s
  10 :   KC  =  000;                                          Counter value, lower limit
  11 :   KC  =  999;                                          Counter value, upper limit
  12 :   KY  =  000,001;                                      Byte or address of a DB
  13 :   KY  =  255,255;                                      Byte or address of a DB
  14 :   KM  =  00000000 00000000;                            Bit pattern, lower limit value
  15 :   KM  =  11111111 11111111;                            Bit pattern, upper limit value
  16 :   KS  =  'Text line with 24 chars";                    ASCII characters enclosed in  ' '
  28 :   S   =  'ENDE';                                       ASCII characters enclosed in  ' '
  30 :
```

Figure 3-25 Input field of the DB Editor

(5) Title field Here, you can enter a maximum 32-character long title for the data block.

(6) DW number field This displays the number of the data word (DW). If the format involves several DWs, the number of the lowest DW is displayed. You can jump to the last data word of the data block by selecting the last DW number or a number higher than the last DW number.

(7) *":" field* Both at this point and in the format field you can insert or delete lines using the function keys. If you delete a line, the whole line including the comment is deleted. When you exit the line with the cursor, all following DW numbers are updated.

(8) Repetition factor field

With the repetition factor, you can reproduce 1 up to a maximum of 12 DWs with the same format. The repetition factor specifies how often the marked block of data words will be entered in the DB. The highest possible repetition factor is 255. All the data words up to and including the cursor position are repeated. The following DW numbers are updated automatically. Data word comments are not reproduced, they remain in their old position.
Before executing a repetition factor, the DB editor checks whether the number of DWs to be reproduced plus the existing DWs will exceed the maximum number of 2043 DWs (without DB header). If this is the case, STEP 5 displays the message :"Memory or internal buffer full". The function is then not executed.

(9) Format field

You input the DW format you require in this field. If the field is already displaying a format, you can overwrite it. If a format cannot be represented, the identifier "**F**" appears in the format error field. If you convert a format that requires several DWs (KG), the next DW is also converted. If several DWs can be represented by a single DW (S, KS) only one DW will be converted.

(10) Editing area

Here, you input data in the current format. If non-interpretable data occur when you change a format, this is indicated in the error field by "**F**".

(11) Format error field

An "F" in this field indicates that an error occurred when interpreting the DW in the specified format.

(12) Comment display field

With data formats requiring several DWs (KS, S, KG), a comment allocated to a DW that is not the first DW cannot be displayed on the screen. A "**C**" indicates these "**suppressed**" comments.

(13) Comment field

Here, you can input a data word comment, if required, for each data word. This is a text up to a maximum of 32 characters long. After the 32nd character, the cursor jumps to the beginning of the comment line again. You can exit the comment line by pressing the ***Return*** key. You can only display "suppressed" comments by changing the data format.

The function keys of the basic menu

F		F	Lib no	F		F	Line fwd
1	Expand DC	2	Delete DC	3	Expand DF	4	Delete DF

F	Line back	F	Title	F	Comment	F	Help
5	KG test	6		7	Enter	8	Cancel

F1
= Expand DC Expand the data word comment; i.e. all the following comment fields are moved one line down.

F2
= Delete DC Delete a data word comment; i.e. all the following comment fields are moved one line up.

F3
= Expand DF Expand a format; i.e. all the following format fields are moved one line down.

F4
= Delete DF Delete a format; i.e. all the following format fields are moved one line up. In the last line of a DB with the format “KG” this function is only executed if you change the format to KM.

F5
= KG test Floating point test. The floating point number in the data field is displayed in hexadecimal form with its exponent (1 byte) and mantissa (3 bytes). This can also be modified. Exit with the ***Insert*** key.

F7
= Enter The data block is stored in the preset program file.

F8
= Cancel End editing without storing.

SHIFT F2
= Lib no Input the library number.

SHIFT F4
= Line fwd Move down one line.

SHIFT F5 *= Line back*	Move up one line.
SHIFT F6 *= Title*	Block title.
SHIFT F7 *= Comment*	Block comment.
SHIFT F8 *= Help*	Display explanation of the function keys.

Structure of a Data Block

A data block created with the DB editor is stored in the preset program file (→ *Project*) and consists of the following parts:

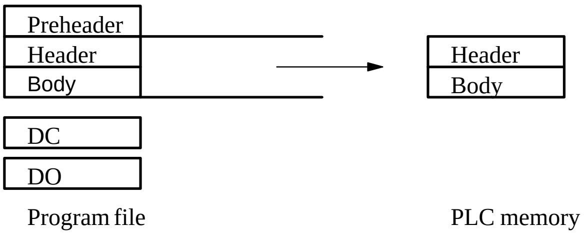
1. Block preheader
2. Block header
3. Block body and if required
4. Comments

When you load the STEP 5 program in the PLC, only the block header (2) and the block body (3) are transferred to the PLC memory.

Block Preheader

The block preheader contains the data formats of the data words in the block body. The length of the preheader depends on the number and order of data formats in the DB. A DVn is generated for a DBn and a DVXn for a DXn. When you delete a DB or DX, its block preheader is automatically deleted along with it.

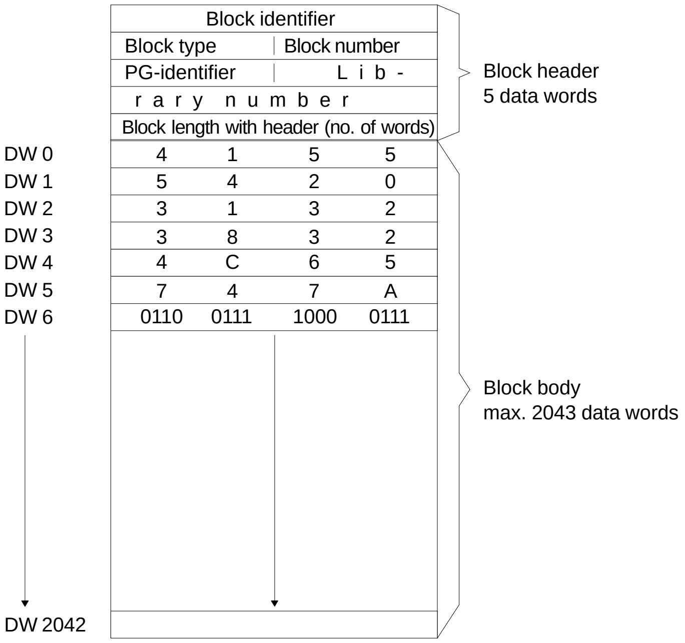
If the block preheader does not exist when you transfer a data block from the PLC memory or EPROM/EEPROM submodule to the preset program file, the following message appears on the screen: “Preheader for this block does not exist”, a line with possible formats is displayed. Using this line, you can set the data format you require.



Block Header

The block header is always 5 data words long. The programmer automatically enters the following information:

- Block start-up ID
- Block type (DB, DX)
- Block number (number between 0 and 255)
- Programmer ID
- Library number (number between 0 and 99999)
- Block length (including the length of the block header)



DW 0	4	1	5	5
DW 1	5	4	2	0
DW 2	3	1	3	2
DW 3	3	8	3	2
DW 4	4	C	6	5
DW 5	7	4	7	A
DW 6	0110	0111	1000	0111

Block Body

The block body contains the data words in ascending order starting with data word DW 0. Each data word takes up one word (16 bits) in the memory. Your user program works with these data words.

DBs created with the DB editor can contain up to 2043 DWs. On the other hand, a data block generated in the user program can contain a maximum of 4091 data words in the block body. The maximum length of a block also depends on the memory capacity of the PLC.

Editing Using Block Comments

Block comments are texts with which you can add information to data blocks. The maximum number of characters of all block comments in a block is 16 K characters. Block comments are stored in a documentation file (DOCFILE) as follows:

– The block and the documentation file are stored in the preset program file. A maximum of 255 documentation files can be stored in one program file under S5-DOS.

– Documentation files are not transferred to the PLC or to an EPROM/EEPROM submodule.

– The number of the documentation file corresponds to the block number, e.g. DBDO.015 belongs to DB 15.

– The documentation files are assigned to the corresponding blocks and preceded by the identifier “#”:
DBn → #DBDO.nnn
DXn → #DXDO.nnn

Note

Use the printer control character **$EJECT** to achieve a form feed. This string must be in upper case letters, otherwise STEP 5 does not recognize the command. If you only write the dollar character, the segment comment after the dollar character is not printed out.

Ready to Start ?

You have selected “*Comments: yes*” in the settings (→ *Project*). The basic menu of the DB editor is displayed on the screen. The DB contains at least one data word.

How to Input Comments

1. Press ***SHIFT F7*** = *Comment* or press the ***COM*** key twice.

   STEP 5 opens the empty editing field for the block comment or displays an existing text. To make sure that the editor can assign the text to the data block, it automatically generates a 7-character string “$1 @”.

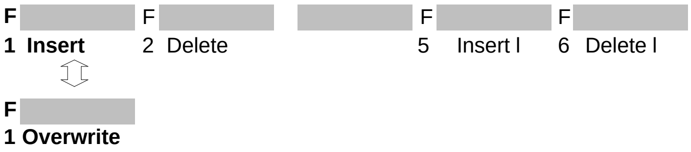
   Do not delete or modify this string, otherwise STEP 5 can no longer identify the block comment as belonging to the particular data block.
2. Edit the text using the alphanumeric keyboard.
3. You can complete each line with the ***Return*** key.

   STEP 5 marks the end of a line with a vertical arrow. If your text covers more than one line, a line break is set automatically.

Inserting Characters

With ***F1*** = *Insert/overwrite* you can change the mode. The selectable mode is always displayed.

1. Position the cursor on the position in the text where you want to insert characters.
2. Press ***F1*** = *Insert* and insert the text.
3. To exit the insert mode: press ***F8*** = *Return* or the ***Insert*** key.



Deleting Characters

1. Position the cursor on the first character to be deleted.
2. Press ***F2*** = *Delete.*
3. Position the cursor after the last character to be deleted.
4. Press ***F2*** = *Delete.*

Completing a Block Comment

*Press **F8** = Return.*
STEP 5 displays the data block to be edited on the screen. The text input up to this point is retained. If you store the data block, STEP 5 then stores the block comment.

Saving the Block Comment

Press the ***Insert*** key.

Inputting the Block Title

The block is identified by the block title. A block title is a maximum of 32 characters long. You can use both upper and lower case letters.

The title is stored in the comment block belonging to the data block. STEP 5 assigns this name automatically (DCn is assigned to DBn). The comment block number is the same as the data block number, e.g. DC 123 belongs to DB 123.

Ready to Start ?

You have selected *“comments : yes”*in the settings (→ *Project*). This basic menu of the DB editor is displayed on the screen. There is at least one data word entered in the DB.

How to Input the DB Title

1. Press ***SHIFT F6*** = *Title* or press the ***COM*** key.
   The cursor jumps to the input field for the block title.
2. Type in the text or correct an existing text.
3. Press the ***Return*** key.
   The title is buffered, but is only stored in the comment block in the program file when you store the whole block.

Influencing the Length of the Block Preheader

The length of the block preheader depends on the number of data formats and their order. If you enter data words with the same format one after the other and avoid changing the data formats too often you obtain a shorter block preheader.

Example

Starting point
The data formats are mixed: DW0/1=KH, DW2/3=KF, DW4=KH and DW5=KF. The block preheader is 10 data words long.

```
DB3           LEN= 11 / 10

0:    KH=   FFFF;
1:    KH=   1A2B;
2:    KF=   + 12345;
3:    KF=    – 00099;
4:    KH=   80F1;
5:    KF=   + 06787;
```

The data formats are grouped together: DW 0 to DW 2=KH, DW 3 to DW 5= KF. The block preheader is now 6 data words long.

```
DB3           LEN= 11 / 6

0:    KH =  FFFF;
1:    KH =  1A2B;
2:    KH =  80F1;
3:    KF=    – 00099;
4:    KF=   + 06787;
5:    KF=   + 12345;
```

When you output data blocks from the PLC, the block preheader must exist in the program file, otherwise STEP 5 displays the message “Preheader does not exist for this block”. In this case, you must select one of the possible formats (KM, KH, KY...).

Inputting the Library Number

The library number is a 5-digit number (0 to 99999) to identify STEP 5 blocks.

Ready to Start ?

The block in which you want to input the library number is open. The DB body must contain at least one DW.

How to Input the Library Number

1. Press ***SHIFT F2*** = *Lib no.*
The cursor is located in the displayed LIB field.
2. Type in the LIB no or modify the existing LIB no.
3. To exit the LIB field, press ***F7*** = *Enter* or the ***Insert*** *key.*

If you want to exit the field without making an entry, press ***F8*** = *Cancel* or ***ESC.***

Changing Data Formats

You can change data formats by positioning the cursor on the format and overwriting it.

Example

You want to change the format in DW 1 to a bit pattern.

1: KH = FFFF;

1. Position the cursor on the format field.
2. Type in the characters ***KM***.

Result:

1: KM = 11111111 11111111;

Inputting Data Words

If the preset program file does not contain a DB with the DB number you have selected, STEP 5 displays the message “Data element does not exist”. You can then start to input data words. If the DB already exists, it is displayed beginning at DW 0.

You can enter a maximum of 2043 data words in a data block (body). If you use formats requiring several data words, STEP 5 displays the lowest data word.

Format	Limit value		Meaning
	lower	upper	
KH	0000	FFFF	Hexadecimal number
KF	–32768	+32767	Fixed point number
KG	–1469368–38	+1701412+39	Floating point number
KT	000.0	999.3	Time value with time base
KC	000	999	Counter value
KY A	000.000	255.255	Byte or address of a DB
KM	00000000.00000000	11111111.11111111	Bit pattern
KS S	ASCII characters, max. 24 characters per line		Text format

The following table shows the number of data words required for the formats.

Format	DWs occupied
KH, KF, KT, KC, KY, KM	1
KG *)	2
KS, S	1 to 12

*) With some negative floating point numbers, rounding errors can occur.

Ready to Start ?

The basic menu of the DB editor is displayed.

How to Input Data Words

1. Type in the required data format in the format field.
   STEP 5 automatically adds the equality sign.
2. Type in the data in the specified data format following the equality sign.

   STEP 5 automatically adds a semicolon, displays the next editing line and repeats the data format you have selected in this line.

The following examples explain how to input different data formats.

Example 1

Hexadecimal numbers:
You want to input KH = **0000** in DW 0 and KH = **FFFF** in DW 1.

1. Type in the characters ***KH***.
   STEP 5 automatically adds the equality sign.
2. Type in the hexadecimal string ***0000***.
   STEP 5 automatically completes the line and displays the next line in the format “KH”.
3. Type in the hexadecimal string ***FFFF***.
   The cursor is now positioned on DW 2.

Example 2

Floating point numbers
You want to enter the floating point number **–0,1469368*10$^{–38}$** in DW 2 and the number **+ 0,1701412*10^{39}** in DW 4.
The cursor is located on DW 2.

```
0:    KH = 0000;
1:    KH = FFFF;
2:    KH = ▮
```

1. Position the cursor on the format field.
2. Type in the character string ***KG***.
3. Type in the numerical strings ***–1469368 –38*** and ***+1701412 +39***

Result:

```
1:    KH = FFFF;
2:    KG = –1469368–38;
4:    KG = +1701412+39;
6:    KG = ▮
```

Example 3

ASCII characters
You want to input the characters **text lines with 24 chars** starting from DW 6 with the format KS and S and **END** in DW 28.
The cursor is positioned on DW 6.

```
4:    KG = +1701412+39;
6:    KG = ▮
```

1. Position the cursor on the format field (**6: ▮**)
2. Type in the characters ***KS***.
3. Type in ***text lines with 24 chars***, the cursor jumps to the next line at DW 18.
4. Overwrite data format KS with S. Type in the characters ***END***.

   The characters “END” are ASCII characters and are not interpreted as the end of the block.

Result:

```
4:    KG = +1701412+39;
6:    KS = ‘text lines with 24 chars‘;
18:   –S = ‘END’ ▮ ;
```

Storing the Block

*Press **F7** or the **Insert** key.*
The data block is stored in the preset program file.

Inputting Data Comments

Data word comments are texts that you can enter in each line of a data format.
You can input data word comments in upper case and lower case letters and they can be up to 32 characters long. Data word comments are stored in the comment block belonging to the data block. STEP 5 assigns the name of the comment block automatically (DCn for DBn). The comment block number is the same as the data block number, e.g. DC 123 belongs to DB 123.

Ready to Start ?

You have selected “*comments: yes*” in the settings (→ *Project*). The basic menu of the DB editor is displayed on the screen. The DB contains at least one data word.

How to Input Data Word Comments

1. Position the cursor on the relevant data word line with ***SHIFT*** and ***cursor right.***
2. Type in a text with a maximum of 32 characters or correct an existing text.

After the 32nd character, the cursor jumps to the start of the comment field.

2. Press the ***Return*** key.

Storing a Comment

The comment block is generated automatically when you first store the data block with comments.
If the comment block already exists, STEP 5 displays the message: “DCn already in destination file, overwrite?”

Press the ***Insert*** key to store the comment.

Reproducing the DWs

With this function you can reproduce a group of DWs (1 to 12 data words of **one** format). The repetition factor "n" specifies how many times the marked data words are required in the DB. You can select a number between 2 and 255 as the repetition factor. When you reproduce a group of data words, you must take into account the maximum data length in a DB (2043 words).

If there are too many data words for a DB, STEP 5 displays "Memory or internal buffer full". The function is not executed.

When you reproduce a group of data words, the original block is included in the reproduced blocks. This means that if you specify n repetitions of the group, on completion of the function, the group exists n times. The DW numbers coming after the reproduced groups are updated.

If you enter a one or two digit repetition factor, you must pad out the number with blanks or type in the character " < " or exit the field using the ***cursor right key***. You then position the cursor in the last format field to be reproduced. The function is executed when you press the ***Return*** key.

Example

You want data words 1 and 2 twice in the DB. The basic menu of the DB editor is displayed on the screen.

Initial situation

0:	KF = +00123;
1:	KH = 8F1A;
2:	KH = 4BBB;
3:	KY = 001,255

Result

0:	KF = +00123;
1:	KH = 8F1A;
2:	KH = 4BBB;
3:	KY = 8F1A
4:	KH = 4BBB;
5:	KY = 001,255;

1. Position the cursor after ***1:*** ▮ with ***SHIFT*** and ***cursor left.***
2. Type in the number ***2***.
3. Move the cursor to the right into the editing field and position it on the number 8 either with **<** and the ***cursor right*** key twice or the ***cursor right*** key four times or the ***space bar*** twice and ***cursor right*** twice.

4. Move the cursor down to the number 4 in DW 2.
5. To reproduce the data words, press the ***Return*** key.

To Cancel the Function

*Press **ESC**.*
If you interrupt the repetition by starting a different operation, the PG displays the message “first finish repetition factor!”. The operation cannot be executed at this point, since the editor is in the repeat mode; this must first be completed.

Testing Floating Point Numbers

Floating point numbers are positive and negative fractional numbers and are represented as an exponential number. You enter the format KG at the PG for floating point numbers. Floating point numbers always occupy a double word (32 bits) in the PLC memory. The mantissa occupies 3 bytes and the exponent 1 byte. If you press the function key **F7** = *KG test* you can display floating point numbers in hexadecimal format and modify them.

Ready to Start ?

The basic menu of the DB editor is displayed on the screen. The DB contains at least one data word.

Example

Testing the floating point number **0,1234567**$^{+12}$ in hexadecimal format.
The floating point number is in data word 1.

```
1:  KG = +1234567 +12
```

1. Position the cursor on the + of the mantissa.
2. Press ***F5** = KG test.*

The number is now displayed in hexadecimal format beside the floating point number, as follows

```
1:  KG = +  1234567+ 12      25            72FA5F
                             Exponent      Mantissa.
```

3. To terminate the display, press ***ESC*** or ***Insert***.

You can change the exponent and mantissa in the hexadecimal format.

4. To enter your changes, press the ***Insert*** key.
5. To discard your changes, press ***ESC***.

Inserting a Line

Using various keys, you can insert or delete DWs and comment lines in a DB.

	Cursor on				
Key	“ : ” field	Format field	Editing area	Comment field	Result
Expand vertical	▨	▨	▨	▨	Line inserted, DW and comment line moved one line down from cursor position.
F3 = Expand DF	▨	▨	▨	▨	Data format inserted, data format moved one line down from cursor position, comments not moved.
F1 = Expand DC				▨	Comment line inserted, DWs not moved, comments moved one line down from cursor position.

Gray shading indicates that the function is possible at this cursor position.

Deleting a Line

	Cursor on				
Key	**“ : ” field**	**Format field**	**Editing area**	**Comment field**	**Result**
Delete key	▨	▨	▨	▨	Data word and comment line deleted, following lines moved one line up.
F4 = Delete DF	▨	▨	▨	▨	Data format deleted, following data formats moved one line up, comments not moved.
F2 = Delete DC				▨	Comment deleted, following comments moved one line up.

Gray shading indicates that the function possible at this cursor position.

Note

If you use ***F3 = Expand DF*** or ***F4 = Delete DF***, the content of the data block can be changed when using the format KG owing to rounding up/down errors.

3.3.6 Editing DB Screen Forms

Editor
DB screen form
in the program file
in the PLC

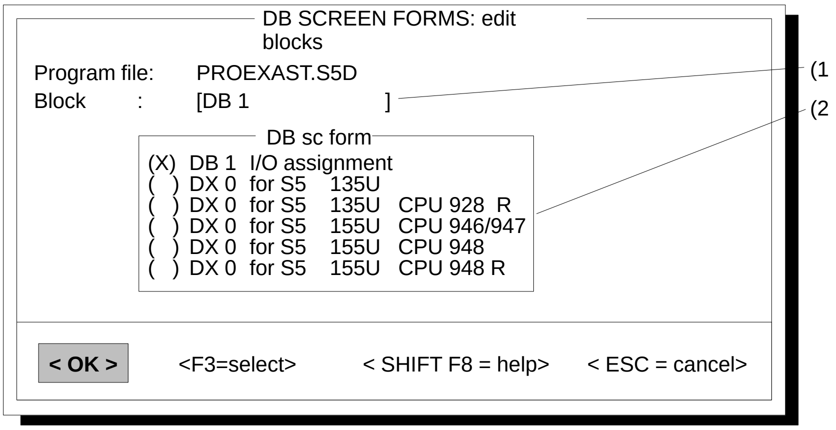
DB screen forms are special data forms for the S5-135U and S5-155U. The parameters you enter depend on the CPU in the PLC. These DB screen forms belong to the particular PLC and do not contain comments.

DB 1 I/O assignment — This contains a list of the digital inputs and outputs (I/Os with relative byte addresses from 0 to 127), IPC flag inputs and outputs for the S5-135U and the timer field length.

DX 0 for the S5-135U — Defaults of certain system program functions for the S5-135U, e.g. for processing the PLC start-up in multiprocessor operation.

DX 0 for the S5-155U — Defaults of some system program functions for the S5-155U, e.g. cold restart, warm restart, process interrupts etc.

Job Box



```
DB SCREEN FORMS: edit blocks

Program file:    PROEXAST.S5D
Block       :    [DB 1                 ]

DB sc form
(X)  DB 1  I/O assignment
(  ) DX 0  for S5    135U
(  ) DX 0  for S5    135U   CPU 928  R
(  ) DX 0  for S5    155U   CPU 946/947
(  ) DX 0  for S5    155U   CPU 948
(  ) DX 0  for S5    155U   CPU 948 R

< OK >     <F3=select>     < SHIFT F8 = help>     < ESC = cancel>
```

Figure 3-26 DB Screen Forms: Edit Blocks

Block

1. Here, you enter the data block in absolute or symbolic form. If you press ***SHIFT and F8*** = *Help* the PG displays a list of possible inputs.
   You can call existing blocks using the block selection box by pressing ***F3*** = *Select*.
2. Here, you select the PLC for which the screen form is intended.

DB1 I/O Assignment for the S5-135U

In multiprocessor operation, each CPU must be assigned digital inputs and outputs, IPC flags and the timer field length. The PG displays a table on the screen in which you can enter these assignments as decimal numbers. The numerical values are stored consecutively in the DB.

Settings for the Editing Session

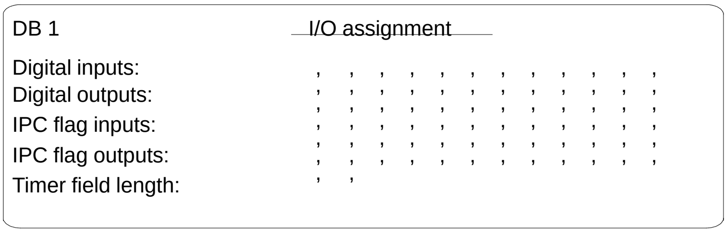
Program file: Name of your current program file.
Mode: ”Online”, if a PLC is connected and you want to edit in the PLC.

For more information about the settings refer to → *Project*.

Selecting the Editor

1. Select the menu editor.
2. Select the DB screen forms in the editor menu.
3. Choose whether you want to edit the block in the program file or in the PLC.
4. Type in the block, e.g. DB 1.
5. Select “DB1, I/O assignment” in the selection box and enter ***DB 1*** as the block.
6. Enter your selections.

The PG displays the I/O assignment screen form below.



```
DB 1                          I/O assignment

Digital inputs:               ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,
Digital outputs:              ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,
                              ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,
IPC flag inputs:              ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,
                              ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,
IPC flag outputs:             ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,  ,
Timer field length:           ,  ,
```

Figure 3-27 I/O Assignment Screen

The feasible and permissible numerical values depend on the configuration of the programmable controller. Refer to the manual of your particular programmable controller.

The cursor is in the first input field of the DB screen.

Inputting the Data

1. Position the cursor in the field in which you want to enter or overwrite a value.

2. Type in the value in decimal.
   After three digits, the cursor automatically jumps to the next field. If you press the ***Return*** key, you jump to the next line.

To Insert a Line

Position the cursor in the line before which you want to insert a line and press the ***expand vertical*** key.

To Delete a Line

Position the cursor in the line you want to delete and press the ***delete segment*** key.

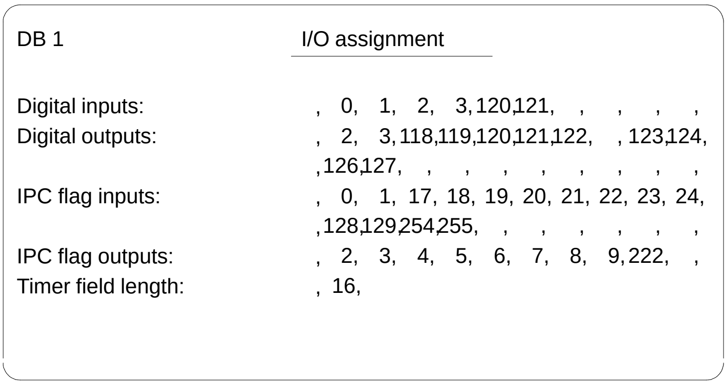
To Delete a Character

Press the ***DEL*** key or overwrite with blanks.

To Enter the Screen Form

Press the ***enter*** key.

Example of a completed DB screen form for the S5-135U:



```
DB 1                           I/O assignment

Digital inputs:                ,  0,  1,  2,  3,120,121,   ,   ,   ,   ,
Digital outputs:               ,  2,  3,118,119,120,121,122,   ,123,124,
                               ,126,127,   ,   ,   ,   ,   ,   ,   ,   ,
IPC flag inputs:               ,  0,  1, 17, 18, 19, 20, 21, 22, 23, 24,
                               ,128,129,254,255,   ,   ,   ,   ,   ,   ,
IPC flag outputs:              ,  2,  3,  4,  5,  6,  7,  8,  9,222,   ,
Timer field length:            , 16,
```

Figure 3-28 Example of a Completed DB Screen

DX 0 for the S5-135U

DX 0 contains the system data for the S5-135U in the form of a DB screen form for this PLC. How to complete the screen form is described in the programming instructions for the PLC.

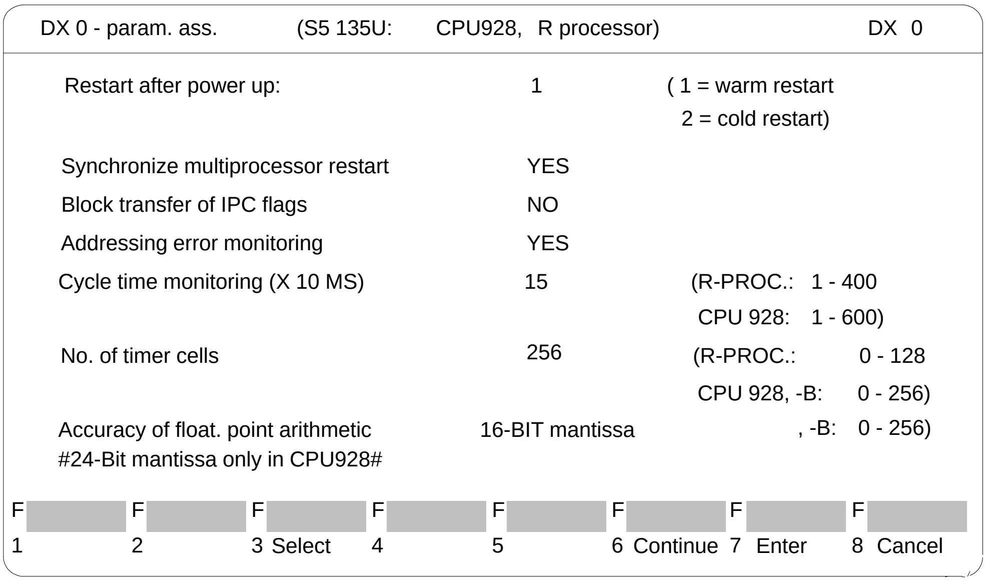
Settings for the Editing Session

Program file: Name of your current program file.
Mode: ”Online”, if a PLC is connected and you want to edit in the PLC.

For more information about the settings refer to → *Project*.

Selecting the Editor

1. Select the editor menu.
2. Select DB screen form in the editor menu.
3. Specify whether you want to edit the block in the program file or in the PLC.
4. Type in the block e.g. DX0
5. Select “DX0 for S5-135U”.
6. Enter your selections with ***OK*** or the ***Return*** key.



```
DX 0 - param. ass.        (S5 135U:    CPU928,  R processor)                DX  0

  Restart after power up:                       1            ( 1 = warm restart
                                                                2 = cold restart)

  Synchronize multiprocessor restart            YES
  Block transfer of IPC flags                   NO
  Addressing error monitoring                   YES
  Cycle time monitoring (X 10 MS)               15              (R-PROC.:   1 - 400
                                                                 CPU 928:   1 - 600)
  No. of timer cells                            256             (R-PROC.:        0 - 128
                                                                 CPU 928, -B:    0 - 256)
  Accuracy of float. point arithmetic        16-BIT mantissa               , -B:  0 - 256)
  #24-Bit mantissa only in CPU928#
```

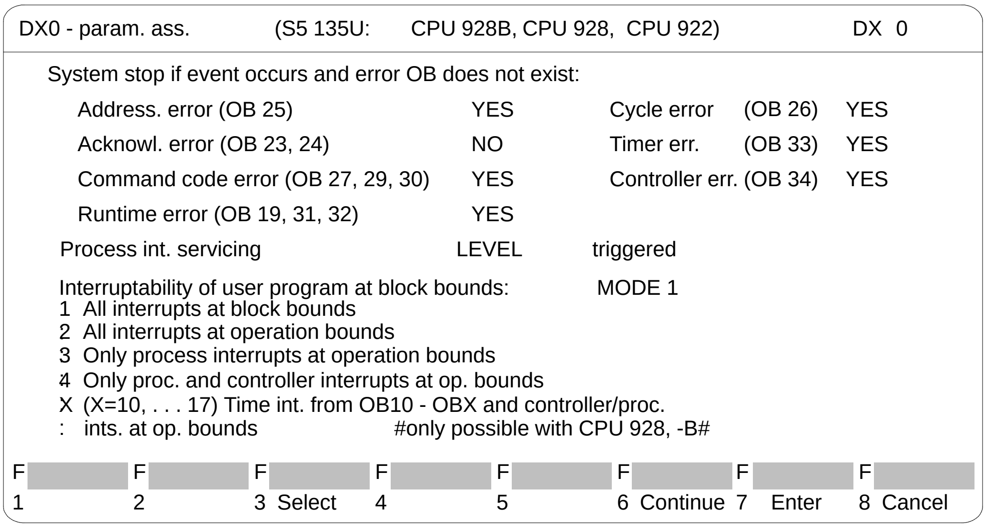
Figure 3-29 DX0 Screen for the S5-135U Page 1

F3 = *Select* Display possible parameters at the cursor position or
F3 = *Input* Input the parameter at the cursor position using the keyboard.
F6 = *Continue* Go to the next page or return to the previous one.
F7 = *Enter* Enter and save the data.
F8 = *Cancel* Return to the previous menu.

The feasible and permissible numerical values depend on the configuration of the programmable controller. Refer to the manual for your particular programmable controller.

Values deviating from the basic setting are displayed red (or on a black background) on the screen. The cursor is located in the first input field of the DX 0 screen form.

DX 0 for S5-135U Page 2:



```
DX0 - param. ass.          (S5 135U:    CPU 928B, CPU 928,  CPU 922)              DX  0

    System stop if event occurs and error OB does not exist:
        Address. error (OB 25)                  YES          Cycle error      (OB 26)   YES
        Acknowl. error (OB 23, 24)              NO           Timer err.       (OB 33)   YES
        Command code error (OB 27, 29, 30)      YES          Controller err.  (OB 34)   YES
        Runtime error (OB 19, 31, 32)           YES
      Process int. servicing                  LEVEL        triggered

      Interruptability of user program at block bounds:         MODE 1
      1  All interrupts at block bounds
      2  All interrupts at operation bounds
      3  Only process interrupts at operation bounds
      4  Only proc. and controller interrupts at op. bounds
      X  (X=10, . . . 17) Time int. from OB10 - OBX and controller/proc.
      :   ints. at op. bounds                  #only possible with CPU 928, -B#

F          F          F          F          F          F          F          F
1          2          3  Select  4          5          6  Continue 7  Enter  8  Cancel
```

Figure 3-30 DX0 Screen for the S5-135U Page 2

Inputting the Data

1. Position the cursor in the field in which you want to change a value.
2. Select the parameter with ***F3*** = *Select* or if ***F3*** = *Input* is displayed, type in the parameter using the keyboard.
3. To call page 2 of the DB screen form, press ***F6*** = *Continue* and type in the parameters as on page 1.
4. To enter DX 1, press the ***Insert*** key or to cancel your input press ***ESC***.

DX 0 for the S5-155U

DX 0 contains the system data for the S5-155U in the form of a DB screen form for this PLC. How to complete the screen form is described in the programming instructions for the PLC.

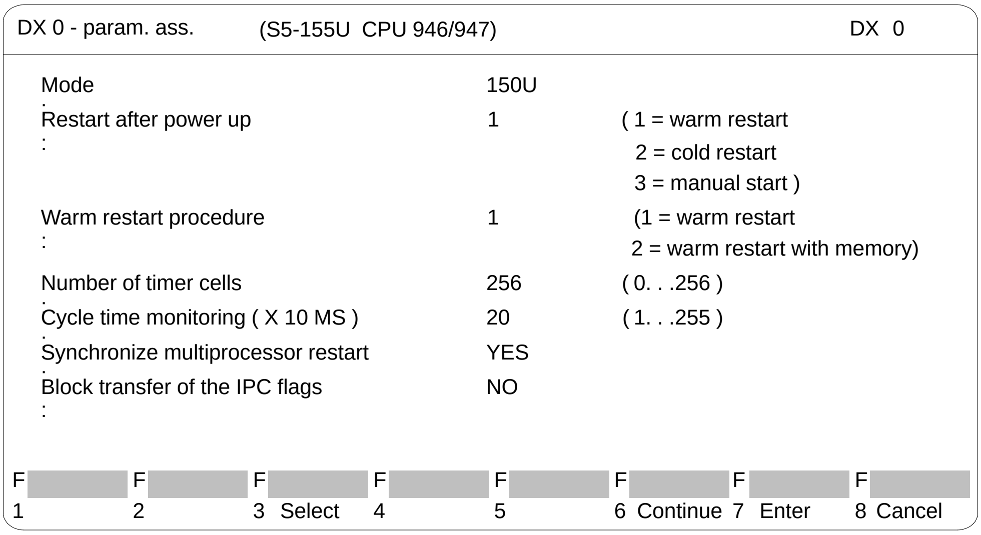
Settings for the Editing Session

Program file:	Name of your current program file.
Mode:	”Online”, if a PLC is connected and you want to edit in the PLC.

Selecting the Editor

1. Select the editor menu.
2. Select DB screen form in the editor menu.
3. Specify wether you want to edit the block in the program file or in the PLC.
4. Type in the block e.g. DX 0
5. Select “DX 0 for S5-155U” in the selection box.
6. Enter your selections.

The PG displays the DX 0 screen form shown below:



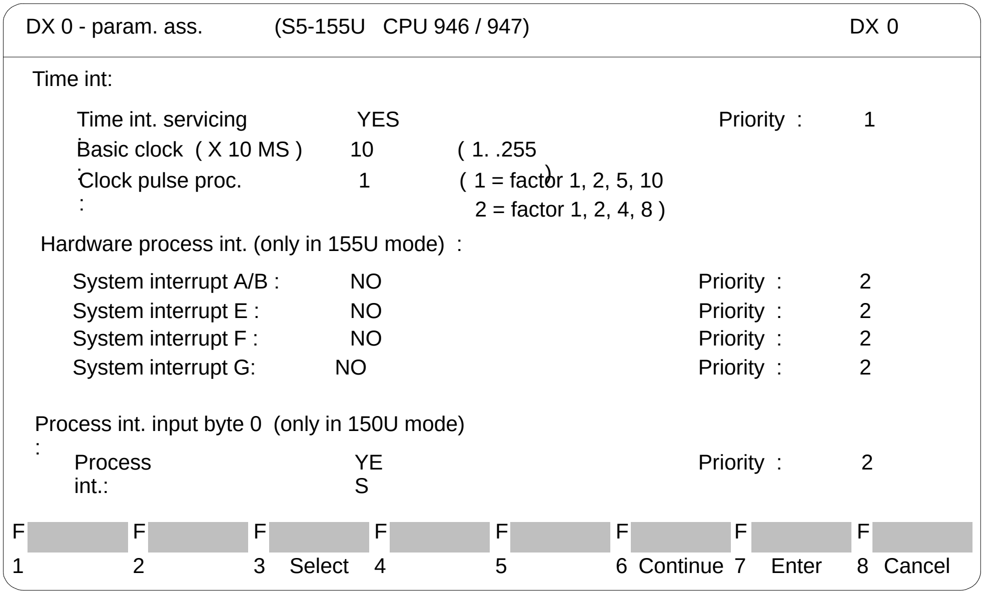
```
DX 0 - param. ass.          (S5-155U  CPU 946/947)                          DX  0

  Mode                                       150U
  :
  Restart after power up                     1            ( 1 = warm restart
  :                                                         2 = cold restart
                                                            3 = manual start )
  Warm restart procedure                     1             (1 = warm restart
  :                                                        2 = warm restart with memory)
  Number of timer cells                      256          ( 0. . .256 )
  :
  Cycle time monitoring ( X 10 MS )          20           ( 1. . .255 )
  :
  Synchronize multiprocessor restart         YES
  :
  Block transfer of the IPC flags            NO
  :
```

Figure 3-31 DX0 Screen for the S5-155U Page 1

F3 = *Select*	Displays possible parameters at the cursor position or
F3 = *Input*	Input the parameter at the cursor position using the keyboard.
F6 = *Continue*	Goes to the next page or returns to the previous one.
F7 = *Enter*	Enter and save the data.
F8 = *Cancel*	Return to the previous menu.

Values deviating from the basic setting are displayed red (on a black background) on the screen. The permitted values depend on the configuration of the programmable logic controller.

DX 0 for S5-155U, page 2:



```
DX 0 - param. ass.          (S5-155U   CPU 946 / 947)                              DX 0

 Time int:

      Time int. servicing           YES                                  Priority  :      1
      Basic clock  ( X 10 MS )     10            ( 1. .255
      :Clock pulse proc.            1            ( 1 = factor 1, 2, 5, 10
      :                                             2 = factor 1, 2, 4, 8 )
  Hardware process int. (only in 155U mode)  :

      System interrupt A/B :       NO                                  Priority  :      2
      System interrupt E :         NO                                  Priority  :      2
      System interrupt F :         NO                                  Priority  :      2
      System interrupt G:        NO                                    Priority  :      2

 Process int. input byte 0  (only in 150U mode)
 :
      Process                      YE                                  Priority  :      2
      int.:                        S
```

Figure 3-32 DX0 Screen for the S5-155U Page 2

Inputting the Data

1. Position the cursor in the field in which you want to change a value, e.g. Mode ***S5-155U*** or ***S5-150U***.
2. Select the parameter with ***F3*** = *Select* or if ***F3*** = *Input* is displayed, type in the parameter using the keyboard.
3. To call page 2 of the DB screen form, press ***F6*** = *Continue* and type in the parameters as on page 1.
4. To enter DX 1, press the ***Insert*** key or, to cancel your input, press ***ESC***.

3.3.7 Parameter Assignment Software COM DB1

Introduction

The COM DB1 parameterization software allows you to assign parameters to CPUs of the low end and mid range. The time required for successful parameterization is minimal.

Until now, it has only been possible to assign parameters to the CPUs in plain text using DB1. Editing of DB1 in plain text was done with the DB editor of the STEP 5 package.

Customer Advantage of Assigning Parameters to DB1 with COM DB1

- COM DB1 can interpret and modify every DB1 with parameterization data and provide it with comments.
- The user need no longer take into account the rules for DB1 parameterization in the PLC manuals since COM DB1 already knows these rules. The user is shown the CPU-specific parameters on the screen. The arguments and the value ranges of the arguments are available in special select boxes.
- COM DB1 can detect input errors in DB1 and indicate these errors in plain text. Errors in DB1 are detected at the latest when transferring to the PLC or to the program file. This excludes the possibility of setting wrong parameters with COM DB1.
- COM DB1 can be used to generate further data blocks required for parameters (e.g. for send and receive mailboxes).
- COM DB1 has online capability, i.e. a generated DB1 can be transferred online to the CPU. In addition, a DB1 can be loaded online from the CPU to the programmer.
- A help text related to the current input can be called up on the screen at any point during parameter assignment.

Scope of Supply of the COM DB1 Software Package

COM DB1 is supplied with STEP 5/ST, V6.6. It is in the directory **\STEP5\S5_COM**.

COM DB1 files:

File name	Contents
s5pxcdbx.cmd	COM DB1 (command file)
s5pdcdbx.dat	Texts in German
s5pecdbx.dat	Texts in English
s5pfcdbx.dat	Texts in French
s5picdbx.dat	Texts in Italian
s5pscdbx.dat	Texts in Spanish

3.3.8 Performance Range of COM DB1

In this chapter, you will learn:

- The functions provided by COM DB1 and the restrictions to be observed
- Which CPUs you can assign parameters with COM DB1.

What Functions does COM DB1 Provide?

The COM DB1 parameterization software is a user-friendly aid for parameterizing low-end and mid-range CPUs.

The functions offered by COM DB1 are described below. Some functions can only be executed with the CPU online. These are indicated specially in the text. All other functions can be used both in online and offline mode. Online and offline mode is selected in the Defaults screen form of COM DB1.

- **Generating a new DB1**

You have just processed a DB1 with COM DB1 and you want to delete it.Press ***F1*** “New DB1” in the “Overview Table” screen form. The DB1 just generated will be deleted and the parameter settings of the default DB1 will appear in the Overview Table.

- **Loading and, if required, modifying a DB1 that already exists in the PLC**

You can modify parameters in a DB1 that already exists in the PLC by selecting “Online mode”, loading the DB1 from the PLC and overwriting the relevant parameters.

- **Loading and, if required, modifying a DB1 that already exist in a STEP 5 program file**

You can modify parameters in a DB1 that already exists in a STEP 5 program file. Select the STEP 5 program file either in the Defaults form or in the “Loading DB1” form. Then load the DB1 from the STEP 5 program file and overwrite the relevant parameters.

- **Generating empty data blocks required for parameter assignment**

(e.g. send mailbox DB for SINEC L1 parameter assignment***)***
When you specify a DB in a parameter block, COM DB1 checks to see if this DB already exists in the PLC (only possible in online mode) or in a STEP 5 program file. If the DB exists but its length is not sufficient for the parameter assignment, the length is corrected.

- **Entering a comment on the entire DB1 or on the current parameter block**

You can enter a comment on the entire DB1 or on the individual parameter blocks. A comment can consist of up to 80 characters (including spaces).

- **Transferring a DB1 to the PLC**

You can transfer a DB1 to the PLC if you have first selected “Online mode”. If there is already a DB1 in the PLC, you will be asked if it is to be overwritten.

- **Transferring a DB1 to a STEP 5 program file**

You can transfer a DB1 to a STEP 5 program file. Specify the STEP 5 program file either in the “Defaults” screen form or in the “Transferring DB1” screen form.

- **Outputting a DB1 to a printer**

You can output DB1 parameters to a printer. All parameter assignment forms and the “Overview table” form can be printed. If you want to use a printer file and/or a footer file for your printout, the printer file or footer file must already exist, i.e. it must already have been generated with the STEP 5 package. You specify the printer file or footer file in the “Defaults” screen form.

- **Outputting a DB1 to a file**

You can output a DB1 to a file. This is necessary if you want to print the DB1 on a printer that is not connected to the programmer. You specify the output file in the “Defaults” screen form.If you want to use a printer file and/or a footer file, the same conditions apply as for direct output of DB1 to a printer. The same contents are output to the file as are output direct to a printer (→ ”Outputting a DB1 to a printer”).

- **Deleting a parameter block**

If you do not want to use parameter blocks, you can delete them in the Overview Table of COM DB1.

- **Executing PLC functions** if you first select “Online mode”:
  - Compressing the PLC memory
  - Switching the PLC from STOP to RUN, the DB1 parameters are updated in the CPU
  - Switching the PLC from RUN to STOP

In addition, COM DB1 provides a range of **Help functions** to make parameter assignment easier for you.

- **Incorrect parameter assignment is prevented** since COM DB1:
  - Detects errors as parameters are entered
  - Examines all inter-parameter dependencies within a DB1
  - Checks that the value ranges of the arguments are not violated
  - Displays an error message in the event of an error and forces you to correct the error (an incorrect DB1 cannot be stored).

Special Features of COM DB1

- COM DB1 can only process one DB1 at a time.
- COM DB1 cannot check the interdependencies of parameters between different PLCs (e.g. whether the same transmission rate is set for all nodes in a SINEC L2 network).
- Direct parameter assignment in the system data is not possible.
- Only those CPU functions which could previously be assigned parameters in DB1 can be assigned parameters with COM DB1.
- If a parameter block in the Overview Table of COM DB1 contains no parameters, the operating system of your PLC automatically writes the available default parameters into the system data.
- Default parameters enclosed between comment characters (#) (→ representation of the default DB1 in the relevant PLC manual) are not recognized by COM DB1 and will be lost. (If the default parameters enclosed in comment characters (#) are immediately in front of the DB1 end-of-text identifier “END”, these characters will be interpreted as comments on the entire DB1.).
- The PLCs listed in Section 3.3.9 can be assigned parameters with COM DB1. The following rules apply to mature PLCs, i.e. same CPU/same PLC with new revision level:

COM DB1 works with the latest PLC revision level known to it, i.e. in the case of a mature PLC, COM DB1 can only assign parameters for the functions it was able to in the last revision level, and it will not recognize any newly added parameters/parameter blocks and/or modified value ranges.

Handling of the individual COM DB1 functions is described in detail in the example of a complete DB1 parameter assignment at the end of this section.

3.3.9 Which PLCs can You Assign Parameters to with COM DB1?

Using COM DB1, you can assign parameters to all the programmable controllers/CPUs listed in the table below:

Programmable controller / CPU	**Can be assigned parameters with COM DB1 from order no. and version**	
S5-90U programmable controller	6ES5 090-8MA01	A01
S5-95U programmable controller::		
♦ Basic unit	6ES5 095-8MA01	A01
♦ with SINEC L2 interface	6ES5 095-8MB01	A01
♦ with two serial interfaces	6ES5 095-8MC01	A01
♦ with SINEC L2-DP interface	6ES5 095-8MD01	A01
S5-100U programmable controller:		
♦ CPU 103	6ES5 103-8MA03	A01
S5-115U programmable controller:		
♦ CPU 941	6ES5 941-7UB11	A01
♦ CPU 942	6ES5 942-7UB11	A01
♦ CPU 943 with one serial interface	6ES5 943-7UB11	A01
♦ CPU 943 with two serial interfaces *****	6ES5 943-7UB21	A01
♦ CPU 944 with one serial interface	6ES5 944-7UB11	A01
and operating system module	6ES5 816-1BB11/21	A01
♦ CPU 944 with two serial interfaces	6ES5 944-7UB21	A01
and operating system module *****	6ES5 816-1BB11/21	A01
♦ CPU 945 with 256 Kbyte memory	6ES5 945-7UA11	A01
and operating system module	6ES5 816-5AA01	A01
♦ CPU 945 with 384 Kbyte memory	6ES5 945-7UA21	A01
and operating system module	6ES5 816-5AA01	A01

Starting COM DB1

With STEP 5 version 6.6 and higher, COM DB1 can be started as follows:

1. Start the S5-DOS/ST operating system (Stage VI).
2. Key on to the “Further SIMATIC S5 programs” start window:

   [Change] [further ...] < ↵ >
3. Make the directory in which you have stored COM DB1 (DR:\STEP5\S5_COM) your working directory.
4. Select the “COM DB1” program in the “Further SIMATIC S5 programs” select box.
5. Start loading COM DB1 with ***OK*** or the ***INSERT*** key.

The “Select Language” screen form (the COM DB1 start screen form) appears on the programmer screen.

3.3.10 How Do You Operate COM DB1?

This Chapter shows you:

- How to proceed when parameterizing with COM DB1 (general operating concept)
- The layout of the COM DB1 screen forms
- How to make entries in the COM DB1 screen forms and the rules to observe in doing so
- The help functions and error messages provided by COM DB1.

Hierarchy of COM DB1 Operator Functions

COM DB1 is operated via screen forms organized into several operator levels. The following applies for all operator levels of COM DB1:

- By pressing one of the function keys ***F1*** to ***F7*** you can execute a COM DB1 function or change to a lower-level COM DB1 screen form.
- You can exit every COM DB1 screen form with the ***F8*** function key and return to the next higher screen form.

The following diagram illstrates the operating concept when working with COM DB1.

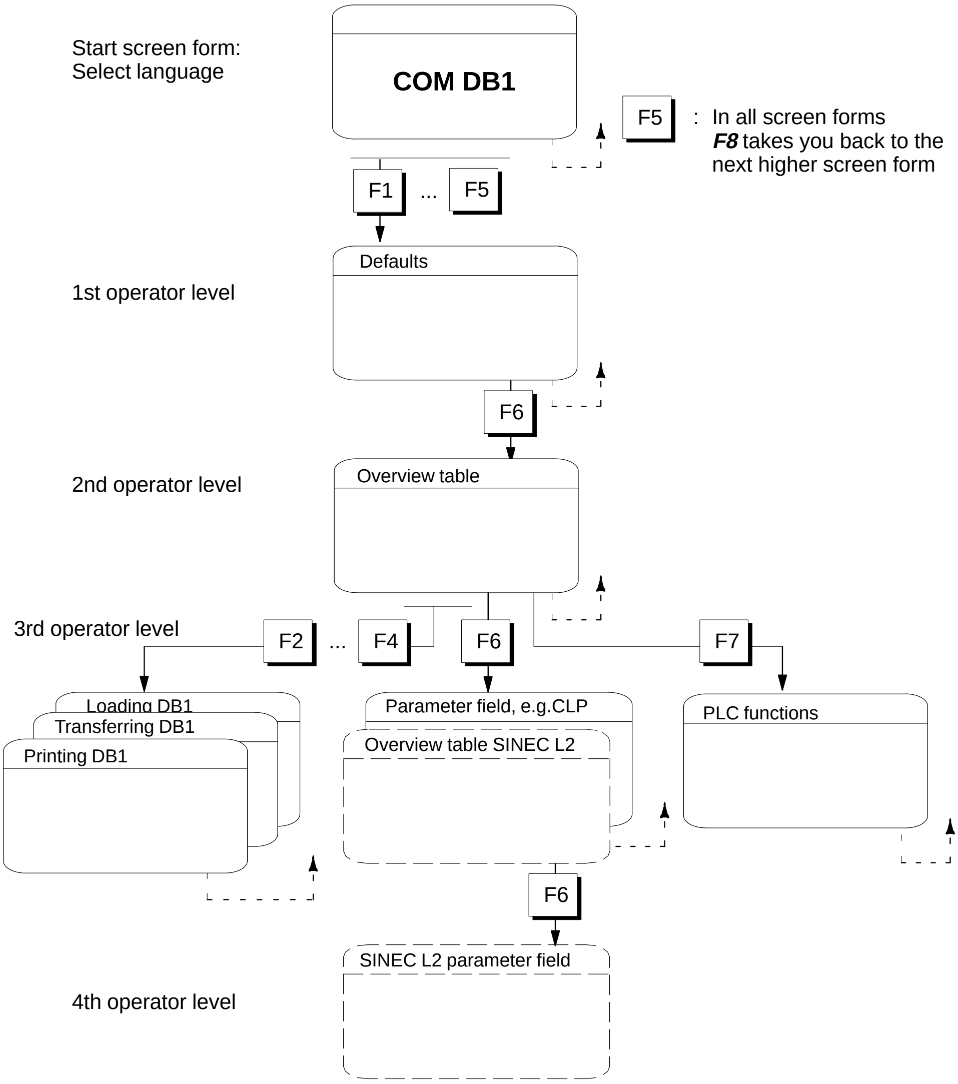


Figure 3-33 Hierarchical Structure of COM DB1

Operating Concept when Assigning Parameters to DB 1 with COM DB1

After starting COM DB1, the first COM DB1 screen form appears. This is the “Select Language” screen form. Use the function keys to select COM DB1 in the desired language.

1st Operator Level: Defaults

After selecting the language, you reach the Defaults screen form. This is where you establish the defaults required by COM DB1 to execute its functions.

You must specify the following in the Defaults screen form:

– Function mode between COM DB1 and the CPU (online, offline)
– Order number of the CPU
– Revision level of the PLC.

Entries in the other input fields of the Defaults screen form depend on the functions you want to execute in the subsequent screen forms. (If, for example, you want to store a DB1 in a program file, you can enter the name of the program file (destination file) in the “Program file:” input field provided for that purpose in the Defaults screen form).

2nd Operator Level: Overview Table

When the defaults have been entered, you reach the “Overview Table” screen form. The Overview Table contains all the parameter blocks possible for the CPU type defined in the Defaults screen form. The “Setting” appears beside each individual parameter block (e.g. “Not parameterized”, “Parameterized (default)”, etc.).

You can decide the following in the “Overview Table” screen form:

– If you want to load, transfer or print a DB1 that exists in the PLC or in a program file (DB1 utility functions)
– If you want to modify or delete parameter blocks of a loaded DB1
– If you want to generate a new DB1
– If you want to branch to a PLC function.

The first time you make the transition from the Defaults screen form to the “Overview Table” screen form, you will be informed in the message line as to whether there is a DB1 in a program file and/or in the PLC. If you load an existing DB1, the Overview Table will be updated.

3rd Operator Level: DB1 Utility Functions/Parameter Block .../PLC Functions

If you have selected a DB1 utility function (e.g. “Loading DB1”) or a PLC function in the 2nd operator level, the relevant screen form for executing the function appears then in the 3rd operator level.

If you have selected a parameter block in the 2nd operator level, you branch to the parameter assignment screen form in the 3rd operator level. The parameter assignment screen form contains a list of all the parameters belonging to the parameter block. Already existing parameter assignment data (e.g. after loading a DB1) appears in the relevant input fields of the parameter assignment screen form. Some input fields without parameter assignment contain default values.

SPECIAL CASE

3. 3rd operator level: SINEC L2 Overview table

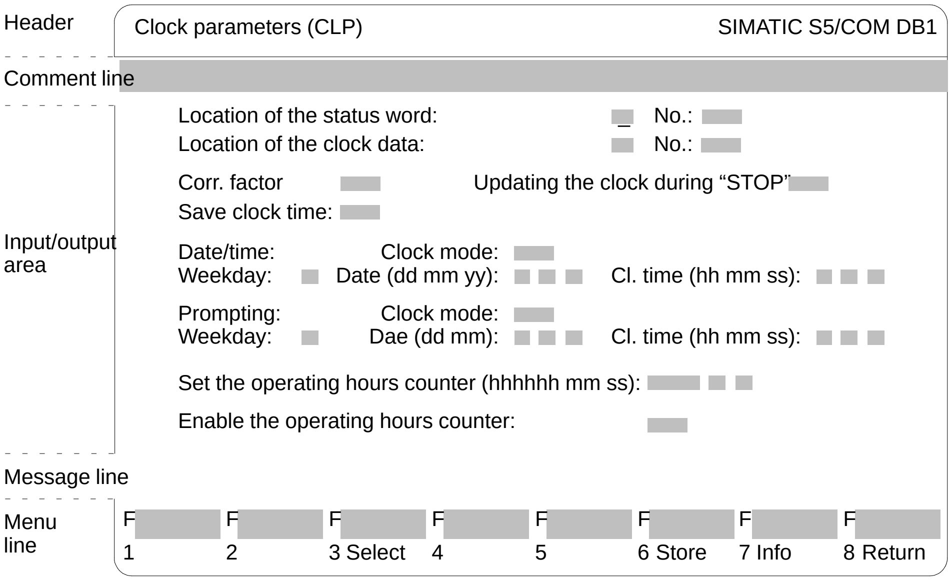
One screen is not sufficient for listing all parameters of the “SINEC L2” parameter block. In this case, the parameter block is divided into logical subunits. After selecting this parameter block in the “Overview Table” screen form, the system takes you to the “SINEC L2 Overview Table” screen form containing the logical subunits.

4th Operator Level: SINEC L2 Parameter Block

The fourth operator level only exists if the “SINEC L2 Overview Table” screen form with the logical subunits appears in the 3rd operator level of COM DB. Each subunit has its own parameter assignment screen form. In the 4th operator level, “SINEC L2 Parameter Block”, the same entries can be made as in the 3rd operator level “Parameter Block...”.

Layout of the COM DB1 Screen Forms

All COM DB1 functions can be executed by entries in screen forms. The COM DB1 screen forms all share the same basic layout. They are divided into five areas. The example below of the “Clock Parameters (CLP)” parameter assignment screen form shows the divisions of COM DB1 screen forms.



Clock parameters (CLP) SIMATIC S5/COM DB1

Location of the status word: _ No.:
Location of the clock data: No.:

Corr. factor Updating the clock during “STOP”
Save clock time:

Date/time: Clock mode:
Weekday: Date (dd mm yy): Cl. time (hh mm ss):

Prompting: Clock mode:
Weekday: Dae (dd mm): Cl. time (hh mm ss):

Set the operating hours counter (hhhhhh mm ss):

Enable the operating hours counter:

F1 F2 F3 Select F4 F5 F6 Store F7 Info F8 Return

Figure 3-34 Layout of the COM DB1 “Clock Parameters (CLP) Screen Form

Header The headers of all COM DB1 screen forms are one line long and separated from the rest of the screen form area by one line. It indicates the contents of the COM DB1 screen form. The header cannot be changed in any COM DB1 screen form.

Comment Line Here you can enter a comment on the parameter block (in the relevant parameter assignment screen form) or on the entire DB1 (in the “Overview Table” screen form). The comment line is one line long and can contain up to 80 characters.

Input/Output Area

The large middle area of the screen is the input area of the COM DB1 screen forms. This area contains fixed texts and input fields, depending on the operator level, in which parameters can be set. Using the keyboard, you can enter the relevant and permissible parameters for the selected function in these input fields and then transfer them to a program file or the PLC.

In the same area, you can view the parameter assignment data of a DB1 existing in a program file or in the PLC (output area). This is also the area where COM DB1 displays select boxes, help windows and warnings for supporting COM DB1 operation.

Message Line

COM DB1 uses the message line to inform you about current processes, operator errors or faults. The first time you make the transition from the Defaults screen form to the “Overview Table” screen form, COM DB1 informs you in the message line as to whether a DB1 exists in a program file and/or in the PLC.

Menu Line

The menu line (function keys ***F1*** to ***F8***) on the bottom edge of the screen tells you which function key on the keyboard executes which COM DB1 function. COM DB1 functions which are not possible in offline mode (e.g. “Load from PLC”) are not supported by the relevant function keys in offline mode.

Possible Entries in COM DB1 Screen Forms and Rules to Observe

This Section shows you:

- How to make entries in the input fields
- How to enter comments in the comment line
- Points to remember when editing.

All inputs to the COM DB1 screen forms are cursor-oriented.

Making Entries in the Input Fields

There are two ways of entering parameter values in the input fields with cursor support:

- 1 Entering the text character-by-character via the keyboard.
- 2 Selecting the text from a select box belonging to the input field (if available) (with ***F3*** "Select").

Note

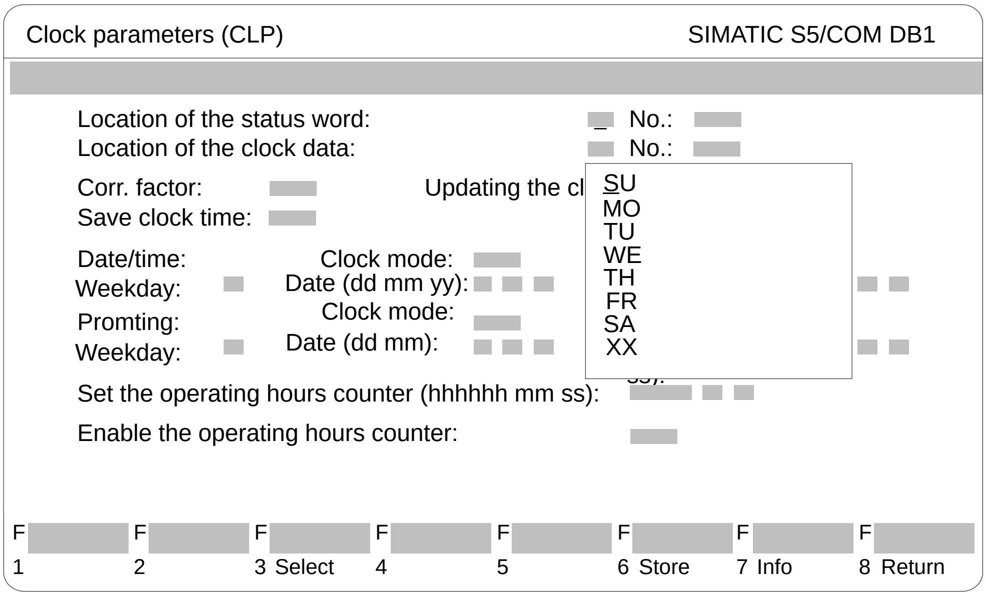
The ***F6*** "Store" key then stores the modified parameter assignment data in DB1. The data is stored only if all parameter assignment data of the block is free of errors. After the data is stored, COM DB1 switches automatically to the "Overview Table" screen form.

Example of 1: Entering a correction factor character-by-character

1. Position the cursor on the "Correction factor:" input field
2. Enter the desired parameter via the keyboard (e.g. "9").
3. Terminate the entry by pressing < ↵ > or <INSERT>. (Press <ESC> to abort the text.)

Example of $\boxed{2}$**:** Entering the day of the week via a select box

1. Position the cursor on the “Weekday:” input field.
2. Open the select box belonging to the input field by pressing ***F3*** “Select”.
3. Position the cursor on the relevant text line in the select box.
4. Enter the selected weekday in the input field by pressing < > or <INSERT>. The selected text appears in the input field. (Press <ESC> to abort the entry.).→ Figure 3-35)



Clock parameters (CLP) SIMATIC S5/COM DB1

Location of the status word: _ No.:
Location of the clock data: No.:

Corr. factor: Updating the cl
Save clock time:

Date/time: Clock mode:
Weekday: Date (dd mm yy):
Promting: Clock mode:
Weekday: Date (dd mm):

ss):

Set the operating hours counter (hhhhhh mm ss):

Enable the operating hours counter:

SU
MO
TU
WE
TH
FR
SA
XX

Figure 3-35 COM DB1 “Clock Parameters (CLP)” Screen Form: Selecting the Weekday

Entering Comments

With COM DB1, you can enter

1. Comments concerning the entire DB1 in the “Overview Table” screen form and
2. Comments concerning each parameter block in the relevant parameter assignment screen form.

You enter the comment in the comment line provided at the top edge of the COM DB1 screen form. The comment can be up to 80 characters long (including spaces).

Example of 2: Entering a comment concerning the “Clock Parameters (CLP)” parameter block

1. Press the <COM> comment key in the “Clock Parameters (CLP)” parameter assignment form. The cursor will then jump to the comment line.
2. Enter the comment via the keyboard (e.g. “Setting the prompter interval of maintenance unit 1”).
3. Terminate the entry by pressing <<Logo3: 1>> or <INSERT>. (Press <ESC> to exit the comment line without changing the original contents.)

Note

A comment concerning a parameter block is stored together with the parameter block (with ***F6*** “Store”) in DB1.

Rules and Points to Remember when Making Entries in COM DB1 Screen Forms

We have collected a few points to remember and rules for parameterizing DB1 with COM DB1 under the “Note” heading below.

Note

- If you do **not** enter the revision level of the CPU in the Defaults screen form, COM DB1 will access the parameter set (parameter blocks, value ranges) of the highest revision level known to it. COM DB1 enters the valid revision level in the relevant input field in the Defaults screen form.
- In the case of CPU 944 with two serial interfaces, you must additionally specify the order number and the version of the operating system module in the Defaults screen form.
- When loading a DB1 generated with STEP 5, comments may be lost if:
  - the comment is longer than 80 characters
  - the comment concerning the entire DB1 is not located immediately in front of the "END" end-of-text identifier
  - the comment concerning a parameter block is not located immediately after the relevant block identifier. Parameter blocks enclosed between comment characters (#) in the default DB1 will also be lost.
- If, before storing a parameter block, you delete a parameter to which a default value has been assigned, the default value remains valid in the PLC. The next time the parameter assignment screen form is selected, the default value appears in the input field of the parameter.

COM DB1 Help and Error Handling Concept

COM DB1 supports you with an extensive help and error handling concept in programming DB1. This Section gives you an overview of the following:

- All the help information which COM DB1 offers during parameter assignment
- All error messages which COM DB1 displays during programming of DB1

Help Concept

The COM DB1 help concept is based closely on the STEP 5 concept.

You can request help texts on the screen depending on the selected COM DB1 screen form and the current cursor position. COM DB1 provides three types of help:

1 Message line: Notes and error messages in the message line of the COM DB1 screen forms

2 Help screen: Help texts with explanations of the current COM DB1 screen form and function key assignments

3 Info window: Help texts with information on the input fields

1 Message line: Notes and error messages in the message line of COM DB1

COM DB1 informs you about the following in the message line of the COM DB1 screen forms:

- COM DB1 operator errors (e.g. “Invalid entry”)
- Parameter assignment errors
- Currently active COM DB1 functions (e.g. “DB1 is being loaded. Please wait...”)
- Existence of a DB1 on a program file and/or in the PLC when changing from the Defaults screen form to the “Overview Table” screen form.

2 Help screen forms: Help texts with explanations of the current COM DB1 screen form and function key assignments

If you press the **<HELP> key** inside a COM DB1 screen form (→ Section 3.5), a help form appears on the screen with a short explanation of the selected screen form and the current function key assignments.

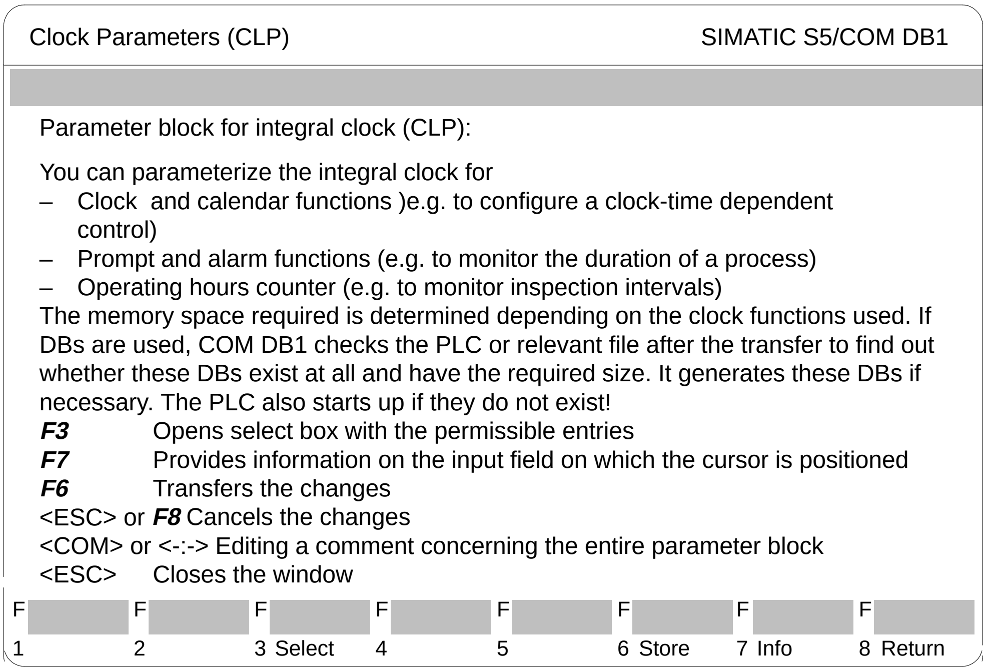
The old screen contents are deleted and the relevant help text is displayed.

If one screen is not sufficient, you can scroll to the next page using the <INSERT> or < ↵ > keys.

Press the **<ESC> key** to exit the help screen form. The old screen contents are restored.

Example:

Help screen form: Explanations of the current COM DB1 “Clock Parameters (CLP)” screen form and function key assignments.



Clock Parameters (CLP) SIMATIC S5/COM DB1

Parameter block for integral clock (CLP):

You can parameterize the integral clock for
- Clock and calendar functions)e.g. to configure a clock-time dependent control)
- Prompt and alarm functions (e.g. to monitor the duration of a process)
- Operating hours counter (e.g. to monitor inspection intervals)

The memory space required is determined depending on the clock functions used. If DBs are used, COM DB1 checks the PLC or relevant file after the transfer to find out whether these DBs exist at all and have the required size. It generates these DBs if necessary. The PLC also starts up if they do not exist!

F3 Opens select box with the permissible entries
F7 Provides information on the input field on which the cursor is positioned
F6 Transfers the changes
<ESC> or ***F8*** Cancels the changes
<COM> or <-:-> Editing a comment concerning the entire parameter block
<ESC> Closes the window

Figure 3-36 Help Screen Form: Explanations of the Current COM DB1 “Clock Parameters” Screen Form and Function Key Assignments

3 Info window: Help texts with information on the input fields

You can request help information concerning the input fields of COM DB1 by pressing function key ***F7*** “Info” (if selectable) . Depending on the cursor position, all possible and permissible inputs are briefly described in an info window.

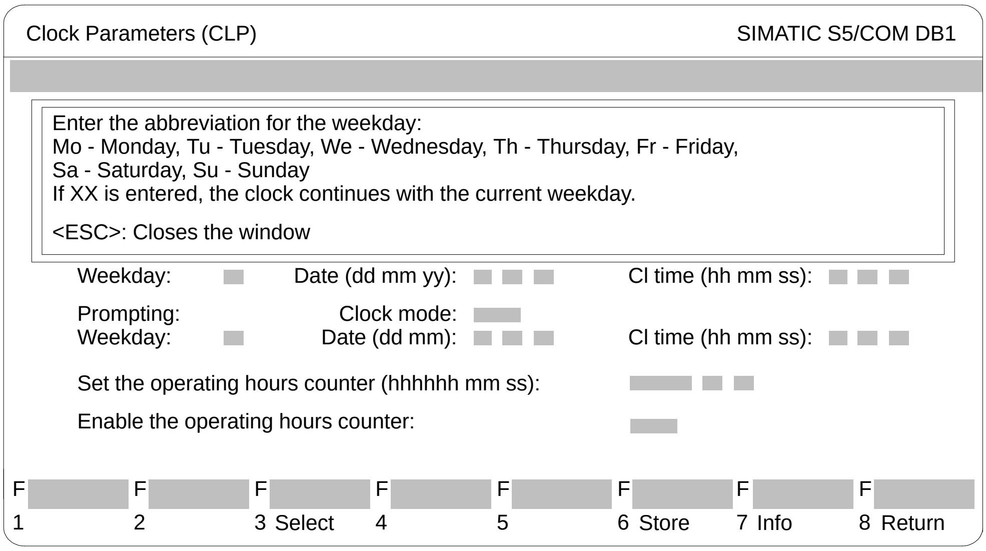
In contrast to the help screen forms, described above, for explaining function key assignments, each info window only overlaps part of the screen so the input field remains visible.

Only one info window can be opened at a time.

The info window must be closed before filling in the input field or positioning the cursor on the next input field. Press the **<ESC> key** to close the info window.

Example:

Info window: information about the “Weekday” input field of the COM DB1 “Clock Parameters (CLP)” screen form.



Clock Parameters (CLP) SIMATIC S5/COM DB1

Enter the abbreviation for the weekday:
Mo - Monday, Tu - Tuesday, We - Wednesday, Th - Thursday, Fr - Friday,
Sa - Saturday, Su - Sunday
If XX is entered, the clock continues with the current weekday.

<ESC>: Closes the window

Weekday: Date (dd mm yy): Cl time (hh mm ss):

Prompting: Clock mode:
Weekday: Date (dd mm): Cl time (hh mm ss):

Set the operating hours counter (hhhhhh mm ss):

Enable the operating hours counter:

Figure 3-37 Info Window: Information about the “Weekday” Input Field of the COM DB1 “Clock Parameters (CLP)” Screen Form

Error Handling Concept

The COM DB1 error handling concept is based closely on the STEP 5 error handling concept. COM DB1 can detect errors and inform the user of them with the relevant messages on the screen.

COM DB1 reacts to the following errors:

1 Errors detected during loading or transferring of DB1

2 Errors during programming of DB1 (operator errors)

COM DB1 reacts to the above listed errors in following ways: either

- With an error message. Error messages are displayed as in STEP 5 in a shortened form in the **message line** on the screen (e.g. “Invalid value range”).
- or with a warning (safety prompt). Warnings are displayed in a plain-bordered **window** in the center of the screen (e.g.: “Do you want to discard the parameter assignment?”). Safety prompts must be acknowledged with <ESC> or answered according to the prompt text with <ESC> for “No, or Abort” or <↵> for “Yes”.

1 Errors detected during loading or transferring of DB1

When loading DB1 from a program file or the PLC, and during transfer of DB1 to the program file or PLC, all parameters are checked for:

– Value range violations
– Parameter dependencies within blocks
– Parameter dependencies between blocks

If COM DB1 detects an error (e.g. “Gaps in input or output area or multiple assignments”), it automatically calls the “Overview Table” in which the parameter blocks concerned are labelled as “errored”:

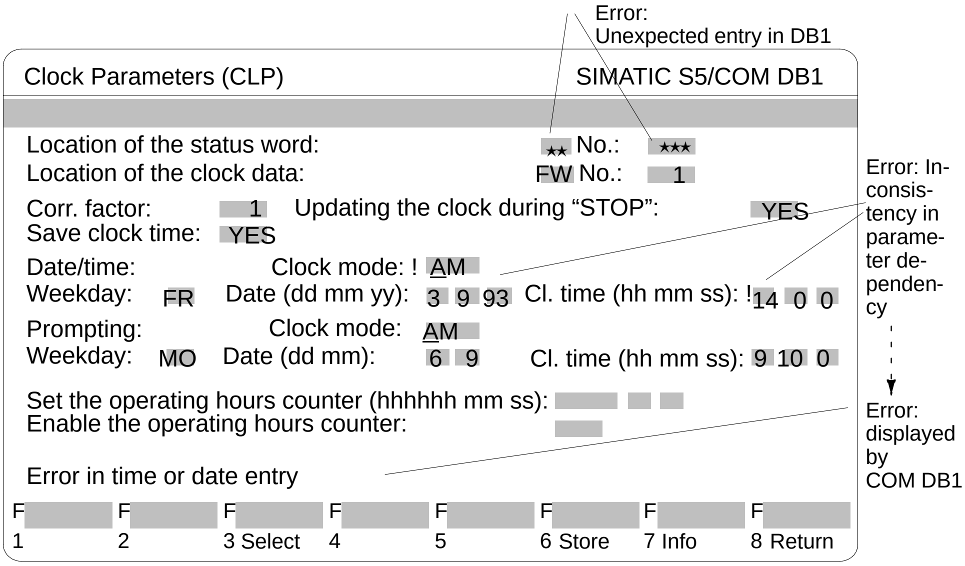
- In the “errored” block, the “genuine” parameter assignment errors are marked with a “**!**” in front of the input field.
- The system enters “*****” in the input field in those cases where data for parameters in the “errored” block cannot be “interpreted” (this can only occur in a DB1 which has been programmed with the DB editor of the STEP 5 package).

Note

If you position the cursor on the erroneous (”!”) parameter in the parameter assignment screen form, the relevant error message will appear in the message line.

Example: Marking erroneous parameters in the “Clock Parameters (CLP)” block after loading DB1. DB1 has been generated with the DB editor of the STEP 5 package.

1. error: “DY” was entered instead of “FY” for the position of the status word. (Typing error, unexpected entry).
2. error: “AM” was entered instead of “PM” for the clock mode. (Wrong value range).



```
Clock Parameters (CLP)                                   SIMATIC S5/COM DB1

Location of the status word:                             ★★ No.:  ★★★
Location of the clock data:                              FW No.:    1
Corr. factor:              1   Updating the clock during “STOP”:        YES
Save clock time:         YES
Date/time:                   Clock mode: ! AM
Weekday:   FR   Date (dd mm yy):  3  9 93  Cl. time (hh mm ss): !14  0  0
Prompting:                   Clock mode:   AM
Weekday:   MO   Date (dd mm):     6  9     Cl. time (hh mm ss):  9 10  0
Set the operating hours counter (hhhhhh mm ss):
Enable the operating hours counter:

Error in time or date entry
F        F        F         F        F        F        F        F
1        2        3 Select  4        5        6 Store  7 Info   8 Return
```

Error:
Unexpected entry in DB1

Error: Inconsistency in parameter dependency

Error:
displayed by
COM DB1

Figure 3-38 Display of Incorrect Parameters in the Parameter Assignment Screen Form

2 Errors during programming of DB1 (operator errors)

Impermissible user inputs are blocked by COM DB1 during programming:

- The input texts are checked by COM DB1 after the entry has been terminated with < ↵ >:
  The user is informed of syntax errors or value range violations with an **error message** e.g. “Invalid value range”). Erroneous parameters are indicated by a **“!”** in front of the input field.
- When the parameter assignment data is stored in DB1 with ***F6*** “Store”, additional parameter dependencies within the block are checked:
  The user is informed of “unfulfilled” parameter dependencies with the **warning** “The parameter assignments cannot be stored since they still contain errors”. After acknowledging with <ESC>, the erroneous parameter settings found in this way are indicated with a **“!”** in front of the input field.

Note

If you position the cursor on the erroneous (**”!”**) parameter in the parameter assignment screen form, the relevant error message will appear in the message line.

Note

Only after all parameters have been correctly entered can the parameter block be stored with ***F6*** “Store”.

3.3.11 Example of a Complete DB1 Parameter Assignment with COM DB1

Using a concrete example, this Chapter shows you how to proceed when parameterizing with COM DB1. This chapter is concerned with the handling of COM DB1 and not with the function to be assigned parameters in DB1.

You will find an explanation of the function and its parameters in the relevant PLC manual. You should read the example below and transfer the principle to your specific application.

The table below contains:

- All the steps required to assign parameters to a PLC;
- All the screen forms in which these steps are executed. (We have included the S5-95U with integral SINEC L2 interface specially for our example).

The individual steps will appear as subtitles in this Chapter.

Table 3-7 *Overview of Procedure for Assigning Parameters to a PLC with COM DB1*

Steps to be Executed in the Following Order and...	Screen Forms Required
1. Install COM DB1	
2. Start COM DB1	
3. Select language	“Select Language” screen form
4. Enter defaults	“Defaults” screen form
5. Switch PLC from RUN to STOP	“PLC Functions” screen form
6. Load Default DB1 from PLC; Enter comment concerning DB1; Select parameter block	“Loading DB1” screen form
7. Enter comment concerning parameter block	“SINEC L2 Overview Table” screen form

Table 3-7 *Overview of Procedure for Assigning Parameters to a PLC with COM DB1*

Steps to be Executed in the Following Order and...	**Screen Forms Required**
8. Edit parameters	”Basic Parameters” screen form ”Standard Connection” screen form
9. Output DB1 to printer	”Printing DB1” screen form
10. Transfer DB1 to PLC	”Transferring DB1” screen form
11. Save DB1 to STEP 5 program file	”Transferring DB1” screen form
12. Switch PLC from STOP to RUN	”PLC Functions” screen form

Description of example task:

An S5-95U with integral SINEC L2 interface is to be assigned parameters. The S5-95U is to communicate with another PLC via the standard connection.

The standard connection is assigned parameters with COM DB1 as described below.

(The parameters and their arguments are taken from the DB1 parameter assignment example for the standard connection in the “SINEC L2 Interface of the S5-95U Programmable Controller” Manual.)

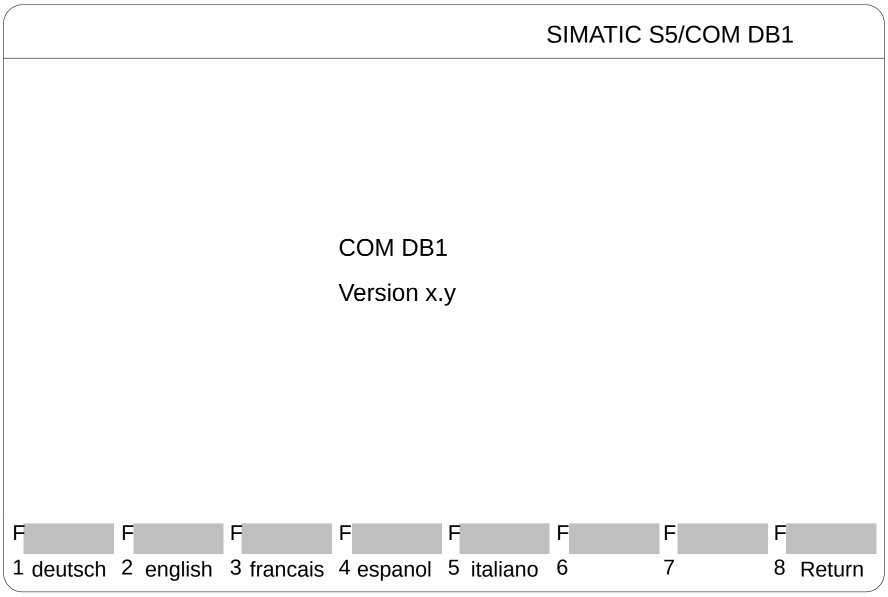
Prerequisites for the example:

- An S5-95U with SINEC L2 interface (Order No.: 6ES5 095-8MB12, Version 01).
- A PG 7XX programmer plugged into the programmer port of the S5-95U.
- The bus connector must not be plugged into the SINEC L2 interface.
- The S5-95U must be in RUN.
- You have generated a program file “AG95L2ST.S5E” with the STEP 5 package.
- You have generated a printer file or footer file with the STEP 5 package.

Selecting the Language

After starting COM DB1, the “Select Language” screen form appears. Use keys ***F1*** to ***F5*** to select the language in which COM DB1 is to appear on the screen.

- Press ***F2*** “English”. (You can exit COM DB1 by pressing ***F8*** “Return” or the <ESC> key.)



SIMATIC S5/COM DB1

COM DB1

Version x.y

Figure 3-39 COM DB1 “Select Language” Screen Form

Defining Defaults

You define the defaults for parameter assignment with COM DB1 in the “Defaults” screen form as described below.

Defining the operating mode between COM DB1 and the CPU:

After selecting the Defaults screen form, the cursor is positioned at the “Online/Offline:” input field.

1. Press ***F3*** “Select” to open the select box belonging to the “Online/Offline:” input field.
2. Press < ↵ > or <INSERT> to enter “Online” in the input field. “Online” appears in the input field.
3. Press < ↵ > or <INSERT> to position the cursor on the next input field.

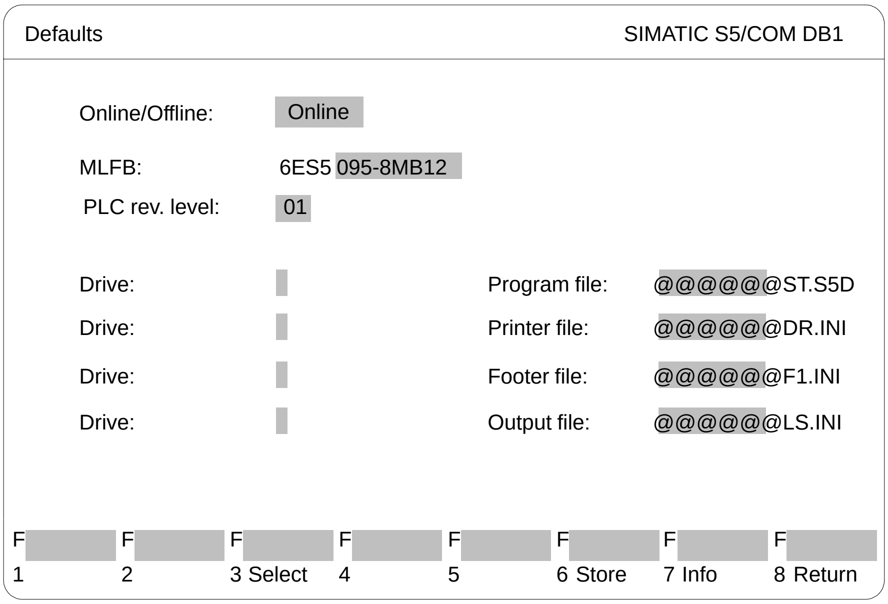
Defining the order number:

1. To define the order number, proceed exactly as you did for “Defining the operating mode between COM DB1 and the CPU”. (You can position the cursor on either the line ”095-8MB22” or “095-8MB02” in the select box.)

Defining the PLC revision level:

2. Enter PLC revision level “01” via the keyboard and terminate the entry by pressing <<Logo3: 1>> or <INSERT>. (You can abort the entry with <ESC>, i.e. the input field will be empty again.)

When you have entered all defaults, the screen form will look like this:



Defaults SIMATIC S5/COM DB1

Online/Offline: Online

MLFB: 6ES5 095-8MB12

PLC rev. level: 01

Drive:		Program file:	@@@@@@ST.S5D
Drive:		Printer file:	@@@@@@DR.INI
Drive:		Footer file:	@@@@@@F1.INI
Drive:		Output file:	@@@@@@LS.INI

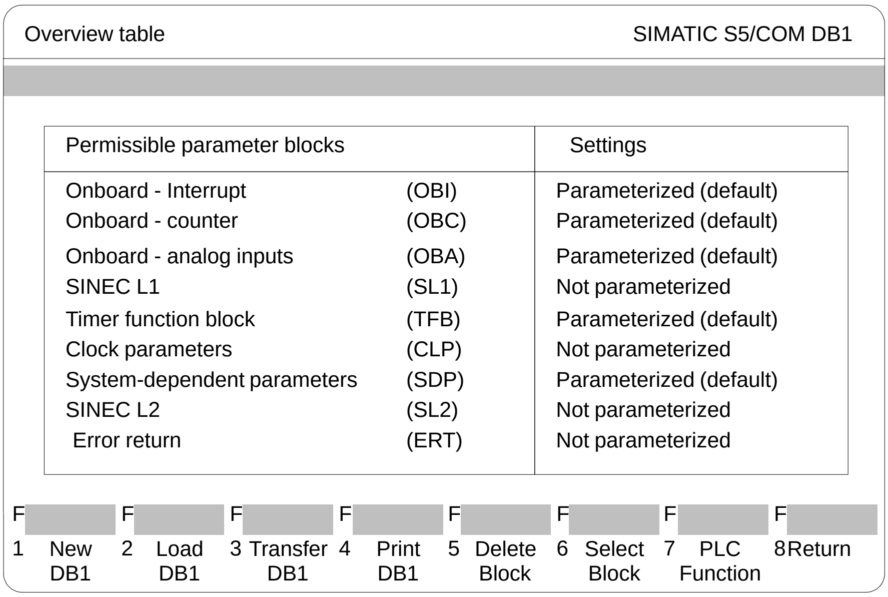
Figure 3-40 COM DB1 “Defaults” Screen Form

3. Store the entries by pressing ***F6*** “Store”. The “Overview Table” screen form appears.

Switching the PLC from RUN to STOP

COM DB1 knows the possible parameter blocks and parameter settings in the default DB1 for the PLC entered in the “Defaults” screen form.

COM DB1 generates the following screen form for the S5-95U:



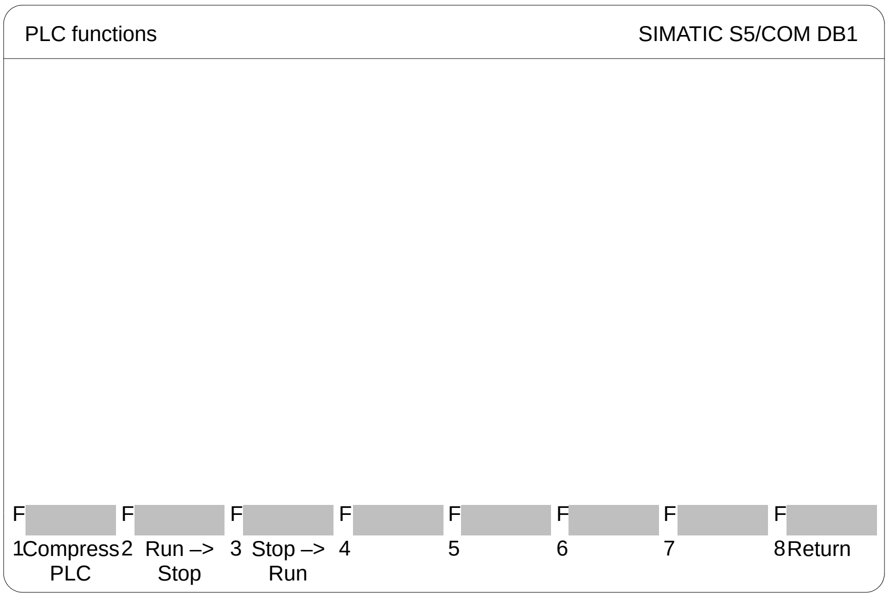
Overview table SIMATIC S5/COM DB1

Permissible parameter blocks		Settings
Onboard - Interrupt	(OBI)	Parameterized (default)
Onboard - counter	(OBC)	Parameterized (default)
Onboard - analog inputs	(OBA)	Parameterized (default)
SINEC L1	(SL1)	Not parameterized
Timer function block	(TFB)	Parameterized (default)
Clock parameters	(CLP)	Not parameterized
System-dependent parameters	(SDP)	Parameterized (default)
SINEC L2	(SL2)	Not parameterized
Error return	(ERT)	Not parameterized

Figure 3-41 COM DB1 “Overview Table” Screen Form

You can change the operating mode of the PLC in the “PLC Functions” screen form:

1. Press ***F7*** “PLC function” for this purpose.
2. Change the operating mode by pressing ***F2*** “Run → Stop”. The PLC is now in STOP.



PLC functions SIMATIC S5/COM DB1

Figure 3-42 COM DB1 “PLC Functions” Screen Form

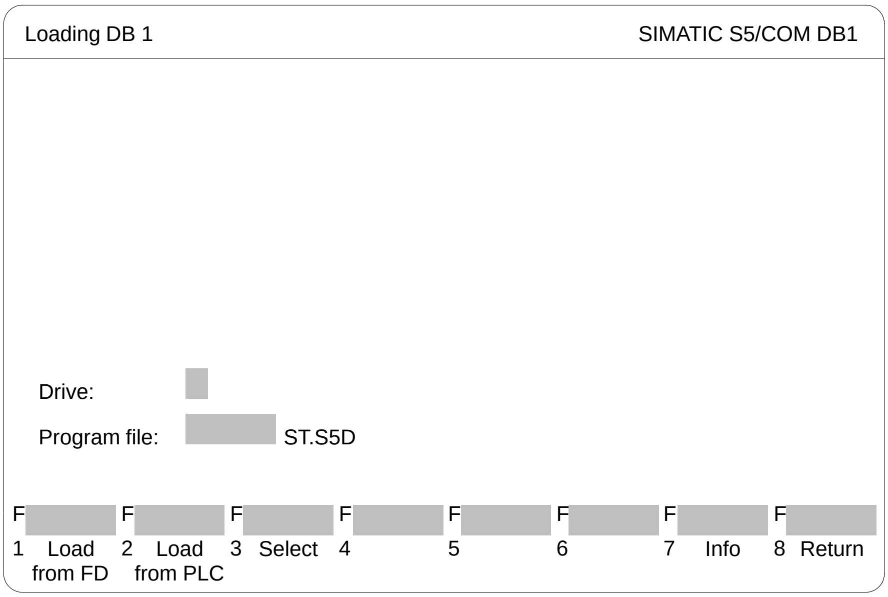
3. Press ***F8*** “Return” to return to the “Overview Table” screen form.

Loading the Default DB1 from the PLC; Entering Comments for DB1; Selecting the Parameter Block

The DB1 in the PLC is to be loaded into COM DB1 and modified.

Loading DB1 from the PLC:

1. Press ***F2*** “Load DB1” in the “Overview Table” screen form (→ *Figure* 3-41). The “Loading DB1” screen form appears as shown below:



Drive:

Program file: ST.S5D

Figure 3-43 COM DB1 “Overview Table” Screen Form

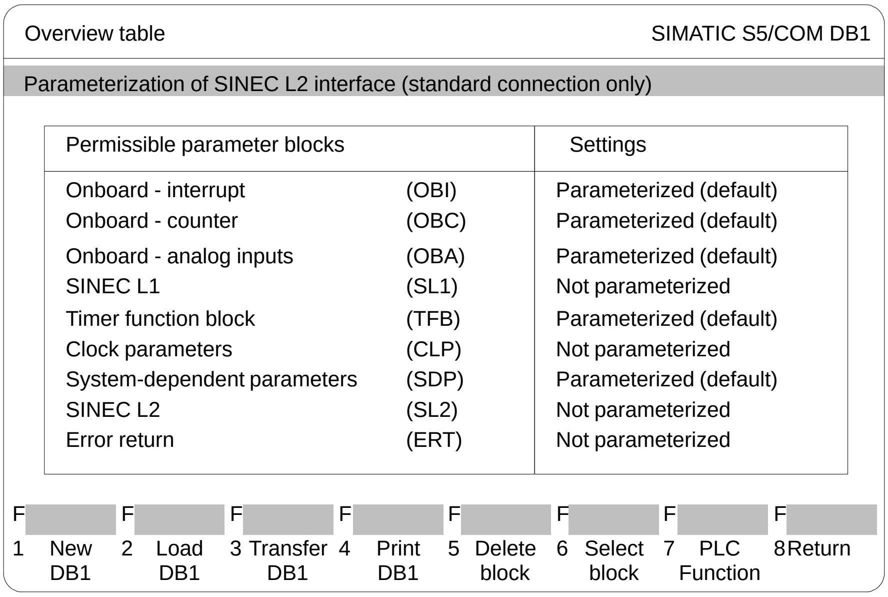
2. Press ***F2*** “Load from PLC”.
   When loading is completed the parameter settings of DB1 in the PLC will be displayed in the Overview Table. Since you have not yet set any parameters in DB1 of the PLC, the default DB1 will be displayed (→ *Figure* 3-44).

Entering a Comment for DB1:

1. If you want to enter a comment, press the <COM> key. The cursor will now be in the comment line of the “Overview Table” screen form.
2. Enter the comment, consisting of up to 80 characters; for our example: “Parameterization of SINEC L2 interface (standard connection only)”
(→ Figure 3-44).
3. Press either < ↵ > or <INSERT>. The cursor then appears in the first line of the “Permissible parameter blocks”.

Selecting the parameter block:

1. To select the parameter block, position the cursor on the parameter block “SINEC L2” (cursor control → Section 3.5).



Overview table SIMATIC S5/COM DB1

Parameterization of SINEC L2 interface (standard connection only)

Permissible parameter blocks		Settings
Onboard - interrupt	(OBI)	Parameterized (default)
Onboard - counter	(OBC)	Parameterized (default)
Onboard - analog inputs	(OBA)	Parameterized (default)
SINEC L1	(SL1)	Not parameterized
Timer function block	(TFB)	Parameterized (default)
Clock parameters	(CLP)	Not parameterized
System-dependent parameters	(SDP)	Parameterized (default)
SINEC L2	(SL2)	Not parameterized
Error return	(ERT)	Not parameterized

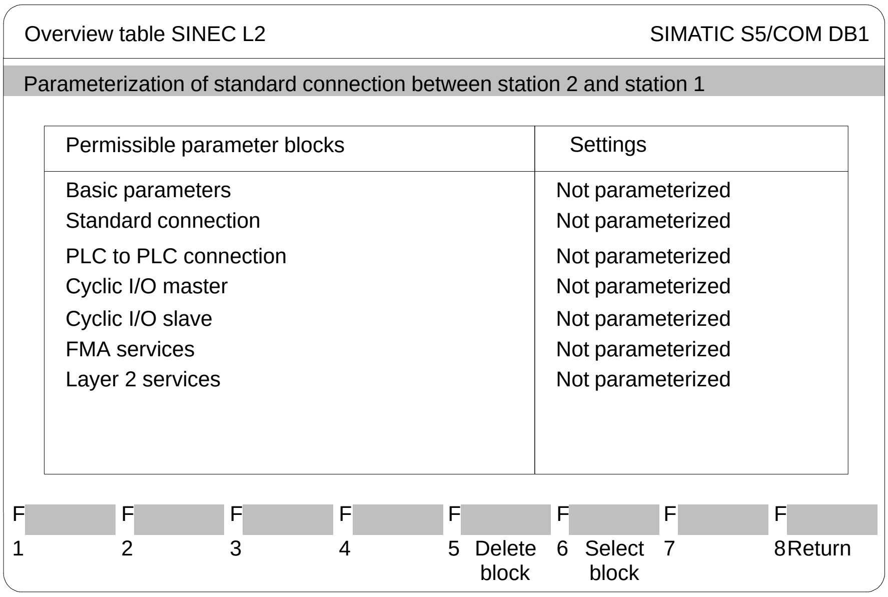
Figure 3-44 COM DB1 “Overview Table” Screen Form

2. Press either <↵ > or <INSERT>. The “Overview table SINEC L2” screen form appears on the screen.

Entering Comments Concerning the Parameter Block

You can enter a comment concerning the SINEC L2 parameter block in the “Overview table SINEC L2” screen form.

1. Press <COM>. The cursor is now in the comment line.
2. Enter the comment consisting of up 80 characters.; for our example: “Parameterization of standard connection between station 2 and station 1”.
3. Press either <↵ > or <INSERT>. The cursor then appears in the line “Basic parameters”.



Overview table SINEC L2 SIMATIC S5/COM DB1

Parameterization of standard connection between station 2 and station 1

Permissible parameter blocks	Settings
Basic parameters	Not parameterized
Standard connection	Not parameterized
PLC to PLC connection	Not parameterized
Cyclic I/O master	Not parameterized
Cyclic I/O slave	Not parameterized
FMA services	Not parameterized
Layer 2 services	Not parameterized

Figure 3-45 COM DB1 “Overview Table SINEC L2” Screen Form

Editing Parameters

In the “Overview table SINEC L2” screen form, you can select the SINEC L2 functions you want to assign parameters to.

Note

You must always define the basic parameters as the first step since these apply to all SINEC L2 functions. Only after this can you define the parameters for the special SINEC L2 functions.

Editing Basic Parameters

Selecting “Basic parameters”:

1. After selection of the “Overview Table” screen form, the cursor is positioned at the “Basic parameters” line.
2. Press either ***F6*** “Store”, <↵ > or <INSERT>. The “Basic parameters” screen form appears (→ *Figure* 3-46).

Defining the station number:

After selecting the “Basic parameters” screen form, the cursor is positioned at the “Station number:” input field.

1. Enter “2” via the keyboard.
2. Store the entry by pressing <↵ > or <INSERT>. The cursor is now at the next input field. (you can abort the entry with <ESC>, i.e. the input field will be empty again.)

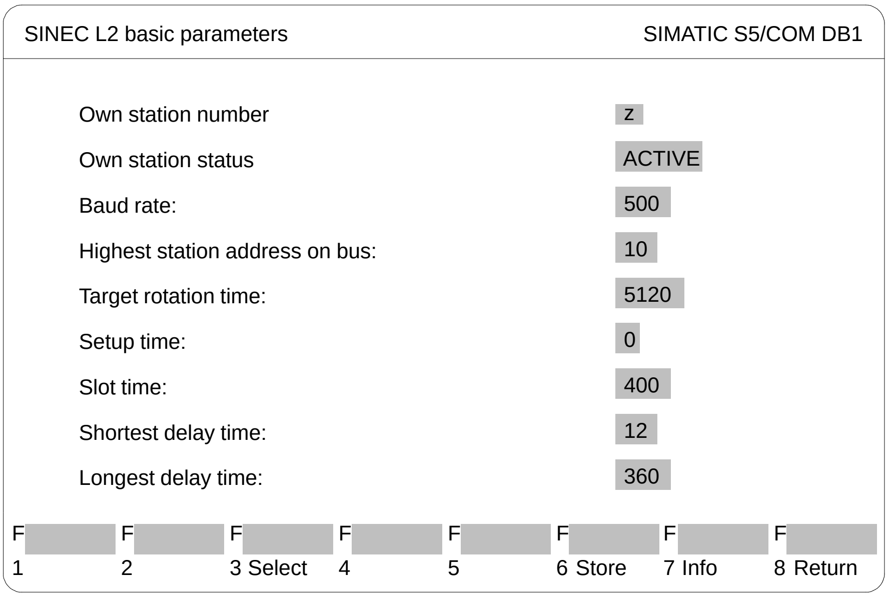
Defining station status:

1. Press ***F3*** “Select” to open the select box belonging to the “Station status:” input field.
2. The cursor is at the “ACTIV(E)” line of the select box.
3. Enter “ACTIV(E)” in the input field by pressing <↵ > or <INSERT>.
4. Position the cursor on the next input field by pressing <↵ > or <INSERT>.

5. Enter all further arguments of the basic parameters as described above:
   - Either direct via the keyboard (you can call up a display of the value range of the arguments via ***F7*** “Info”) or
   - Using the select box.

Please see the screen form in 3-46 for the parameter arguments.

When you have entered all basic parameter arguments, the screen form appears as shown below:



SINEC L2 basic parameters SIMATIC S5/COM DB1

Own station number	z
Own station status	ACTIVE
Baud rate:	500
Highest station address on bus:	10
Target rotation time:	5120
Setup time:	0
Slot time:	400
Shortest delay time:	12
Longest delay time:	360

Figure 3-46 COM DB1 “SINEC L2 Basic Parameters” Screen Form

6. Press ***F6*** “Store”. The basic parameters are stored in DB1 and the “Overview table SINEC L2” screen form appears (→ Figure 3-45). “Parameterized” appears in the “Basic parameters” line in the screen form.

(Press <ESC> or ***F8*** “Return” to abort the entry. The “Overview table SINEC L2” screen form then appears in its original form.)

Editing Parameters for Standard Connection

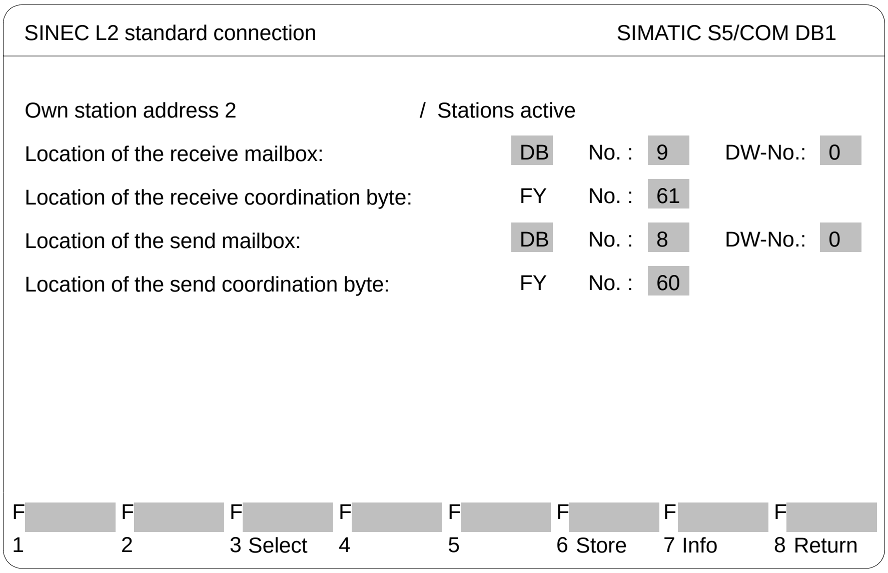
Select “Standard connection”:

The cursor is in the “Overview table SINEC L2” (→ *Figure* 3-45) screen form at the “Standard connection” line.

1. Press either ***F6*** “Store”, <↵ > or <INSERT>. The “SINEC L2 Standard Connection” screen form appears.
2. Enter all parameter arguments as described for the basic parameters either directly at the keyboard or using the select box.

Please see the screen form in Figure 3-45 for the parameter arguments.

When you have entered all the arguments, the screen form appears as shown below:



SINEC L2 standard connection SIMATIC S5/COM DB1

Own station address 2 / Stations active

Location of the receive mailbox: DB No. : 9 DW-No.: 0

Location of the receive coordination byte: FY No. : 61

Location of the send mailbox: DB No. : 8 DW-No.: 0

Location of the send coordination byte: FY No. : 60

Figure 3-47 COM DB1 “Standard Connection” Screen Form

3. Press ***F6*** “Store”. The parameters are stored in DB1 and the “Overview table SINEC L2” screen form appears (→ *Figure* 3-45). “Parameterized” appears in the “Standard connection” line in the screen form.

   (Press <ESC> or ***F8*** “Return” to abort the entry. The “Overview table SINEC L2” screen form then appears in its original form.)

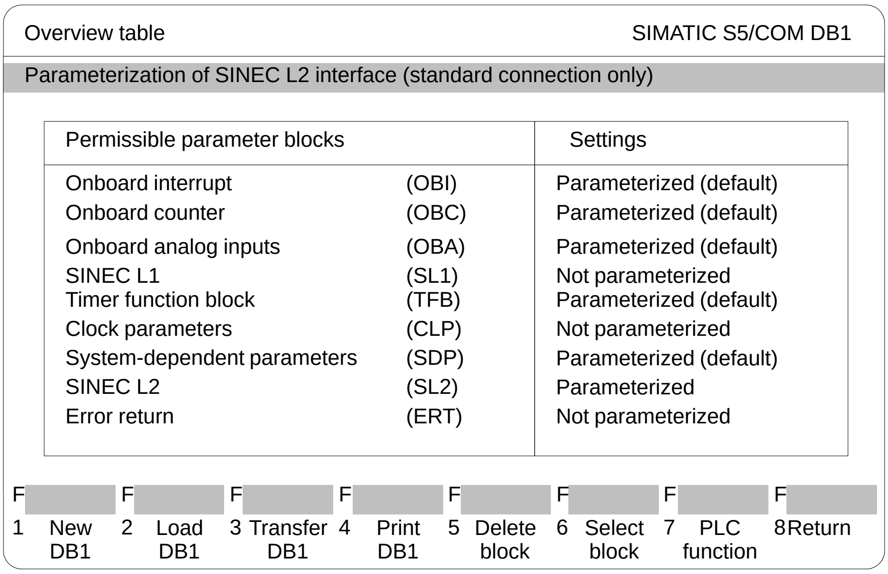
The parameter assignment of example DB1 is now complete.

Outputting DB1 to the Printer

You want to print the DB1 you have just generated.

1. Press ***F8*** “Return” twice to return to the “Overview table” screen form.

The “Overview table” screen form has changed; the SINEC L2 parameter block is displayed as having parameters assigned:



Overview table SIMATIC S5/COM DB1

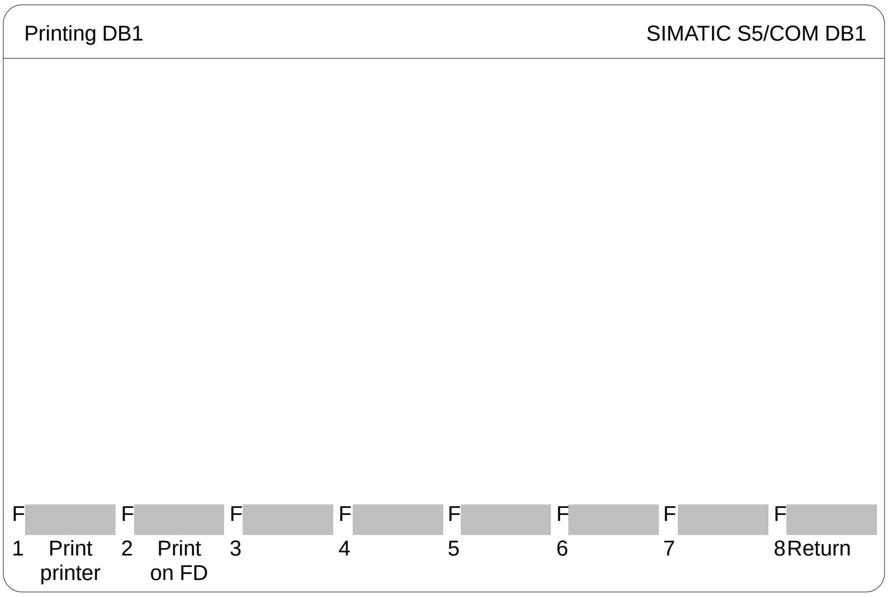
Parameterization of SINEC L2 interface (standard connection only)

Permissible parameter blocks		Settings
Onboard interrupt	(OBI)	Parameterized (default)
Onboard counter	(OBC)	Parameterized (default)
Onboard analog inputs	(OBA)	Parameterized (default)
SINEC L1	(SL1)	Not parameterized
Timer function block	(TFB)	Parameterized (default)
Clock parameters	(CLP)	Not parameterized
System-dependent parameters	(SDP)	Parameterized (default)
SINEC L2	(SL2)	Parameterized
Error return	(ERT)	Not parameterized

Figure 3-48 COM DB1 “Overview Table” Screen Form

2. Press ***F4*** “Print DB1”.

The “Printing DB1” screen form appears as shown below:



Printing DB1 SIMATIC S5/COM DB1

Figure 3-49 COM DB1 “Printing DB1” Screen Form

3. Press ***F1*** “Print printer”..
   This prints the “Overview table” screen form, the “Overview table SINEC L2” screen form and all parameter assignment forms of the SINEC L2 block. The number of the page currently being printed is displayed in the message line.

When printing has been completed, the “Overview table” screen form automatically appears.

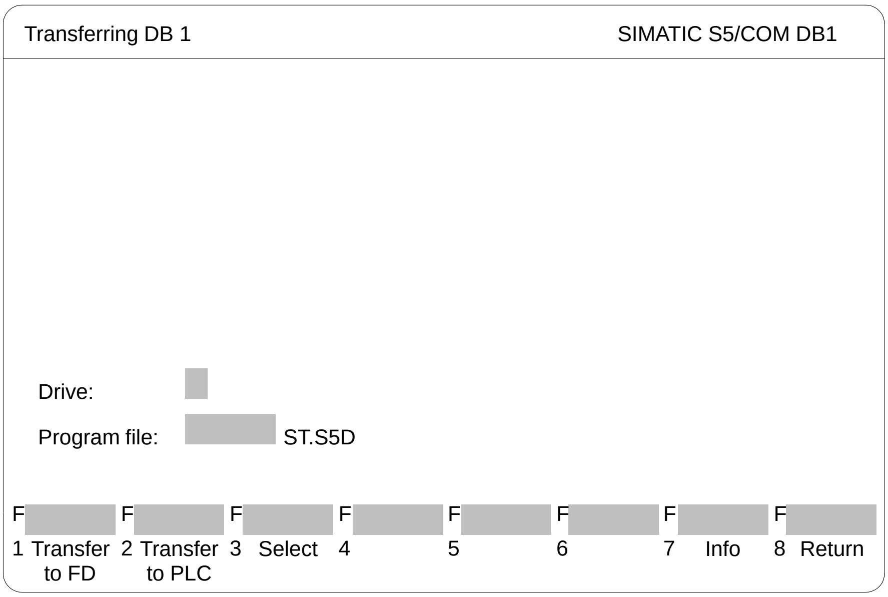
(If DB1 has not been printed, you will receive a relevant message.)

Transferring DB1 to the PLC

The DB1 you have just generated will be transferred to the PLC.

1. Press ***F3*** “Transfer DB1” in the “Overview table” screen form (→ Figure 3-48).

The “Transferring DB1” screen form appears as shown below:



Transferring DB 1 SIMATIC S5/COM DB1

Drive:

Program file: ST.S5D

Figure 3-50 COM DB1 “Transferring DB1” Screen Form

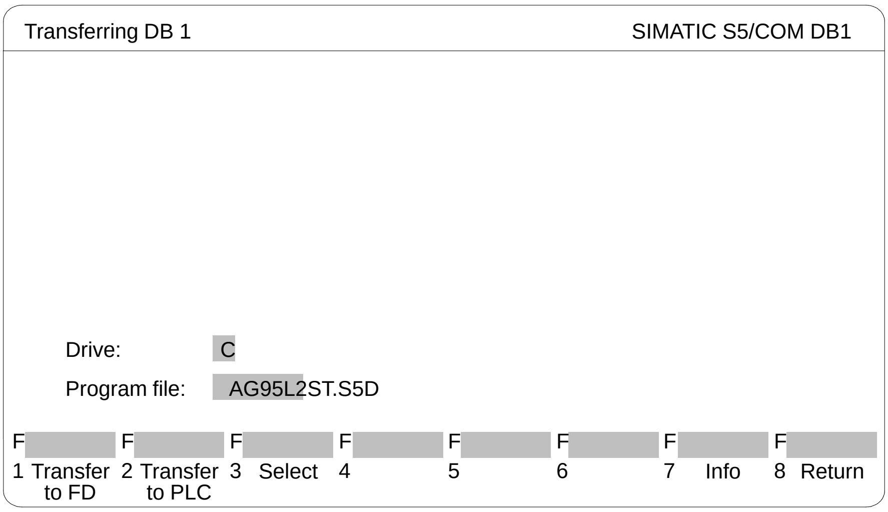
2. Press ***F2*** “Transfer to PLC”. The message line now informs you that DB1 is being transferred. The DB1 in the PLC is simultaneously overwritten.

When transfer of DB1 is complete, the “Overview Table” screen form automatically appears. (If there are errors in DB1, you will get the relevant message and DB1 will not be transferred. The erroneous parameter block will be indicated in the Overview screen form.

Saving DB1 to a STEP 5 Program File

The DB1 you have just transferred to the PLC should be saved/archived to a STEP 5 program file (or diskette). For this purpose, you must specify the STEP 5 program file to which DB1 is to be stored in the "Transferring DB1" screen form. It was a condition for our example that you had already generated the STEP 5 program file "AG95L2ST.S5E" with the STEP 5 package.

1. Press ***F3*** "Transfer DB1" in the "Overview table" screen form (→ *Figure* 3-48). The "Transferring DB1" screen form appears.
2. nter the STEP 5 program file and the drive (→ *Figure* 3-51).



Transferring DB 1 SIMATIC S5/COM DB1

Drive: C

Program file: AG95L2ST.S5D

Figure 3-51 COM DB1 "Transferring DB1" Screen Form

3. Press ***F1*** "Transfer to FD". The message line then informs you that DB1 is being transferred.

When transfer of DB1 is complete, the "Overview table" screen form automatically appears.

(If there are errors in DB1, you will get the relevant message and DB1 will not be transferred. The incorrect parameter block will be indicated in the Overview screen form.

Switching the PLC from STOP to RUN

You can change the operating mode of the PLC in the “PLC functions” screen form.

1. Press ***F7*** “PLC functions” in the “Overview table” screen form (→ *Figure* 3-48). The “PLC functions” screen form appears.
2. Change the operating mode by pressing ***F3*** “Stop → Run”. You will be asked if the parameter settings in the PLC are to be updated.
3. To acknowledge, press <↵ > or <INSERT>. The parameter settings will be transferred to the operating system of the PLC.

   (You can abort updating in the PLC with <ESC> or ***F8*** “Return”.)

The parameter settings in the PLC have been updated and the PLC is in RUN.

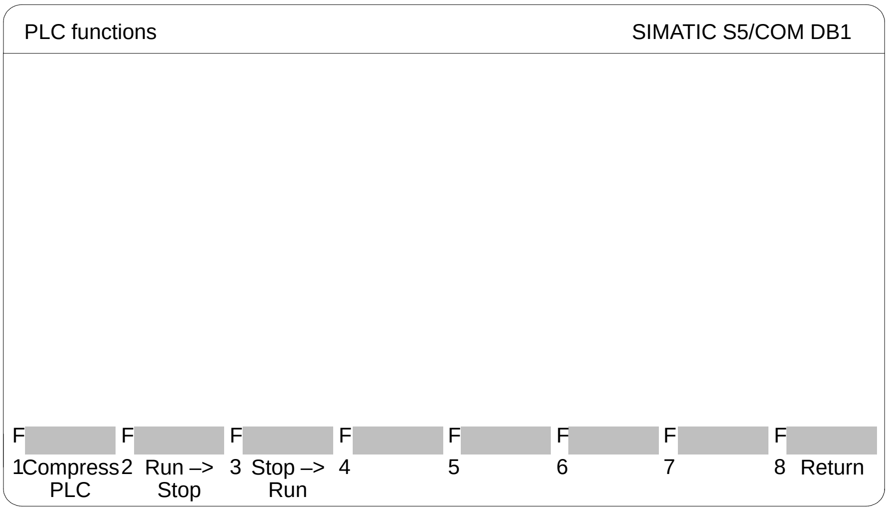


Figure 3-52 COM DB1 “PLC Functions” Screen Form

4. Exit COM DB1 by pressing ***F8*** “Return” 4 times.

3.3.12 Editing the Assignment List

Editor

Assignment list

With symbolic programming, you can specify a string of alphanumeric characters, e.g. BUTTON-ON instead of an absolute operand, e.g. I 1.1. Before you can program with symbolic operands, you must create a list of assignments between the absolute and symbolic operands using the STEP 5 symbolic editor. While making these assignments, you can also write an operand comment for each operand.

You can select the length of the symbolic operands and operand comments (→ *Project, Settings*), as follows:

- symbolic operand: 8 to 24 characters (8 default),
- comment:max. 40 characters (40 default).

These settings are valid for all your work with the assignment list. You can increase the length easily later. You can, however, only decrease the length to that of the longest comment in the file (first delete ?????Z?.INI).

The assignments and modifications to the assignments are made in the sequential source file. After editing, this file is converted to the symbols file (*Z0.INI) when you store the source file.

You must enter the name of the symbols file in the settings. This name is then automatically used for the sequential source file.

If you select the assignment list submenu, the editor for the sequential source file (*Z0.SEQ) is called immediately.

STEP 5 then displays an (empty) assignment list with columns for the following:

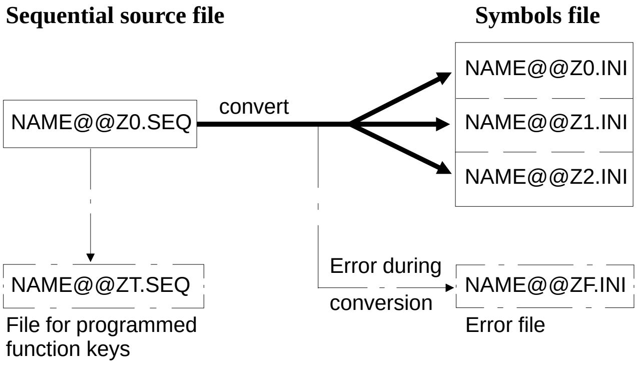
- absolute operands,
- symbolic operands,
- operand comments and
- → *additional comments*, beginning with a semicolon.

To create the assignment list, follow the procedure outlined below.

1. You edit the assignment list as the sequential source file (extension *Z0.SEQ).

2. The sequential source file is translated into the symbols file (three files with the extensions Zx.INI, x = 0, 1, 2) when you store the symbols file. If errors occurred during the conversion, STEP 5 stores the errors in an error file (extension *ZF.INI). You can display or print out this error list with the functions 〈→ *Management*, *Assignment lists*, *Output error list*).

   If you have assigned texts to the function keys for editing the assignment list (→ *Programmable function keys*) the file *ZT.INI is also created.



3. The stored symbols file is used to translate the user program into machine code and for the output.

Permitted Operand Types

The following table lists all the operand types to which you can assign a symbolic name in the assignment list.

Table 3-8 Overview of the Permitted Operand Types

Operand	Explanation	Operand	Explanation
C	Counter	IW	Input word
D	Bit in data word	OB	Organization block
DB	Data block	OW	Word in ext. I/Os
DD	Data double word	OY	Byte in ext. I/Os
DL	Data word, left byte	PB	Program block
DR	Data word, right byte	PW	Peripheral word
DW	Data word	PY	Peripheral byte
DX	Extended data block	Q	Output
F	Flag	QB	Output byte
FB	Function block	QD	Output double word
FD	Flag double word	QW	Output word
FW	Flag word	S	Extended flag
FX	Extended function block	SB	Sequence flag
FY	Flag byte	SD	Ext. flag double word
I	Input	SW	Extended flag word
IB	Input byte	SY	Extended flag byte
ID	Input double word	T	Timer

Note

Variables blocks (VB) can also be assigned a symbolic name.

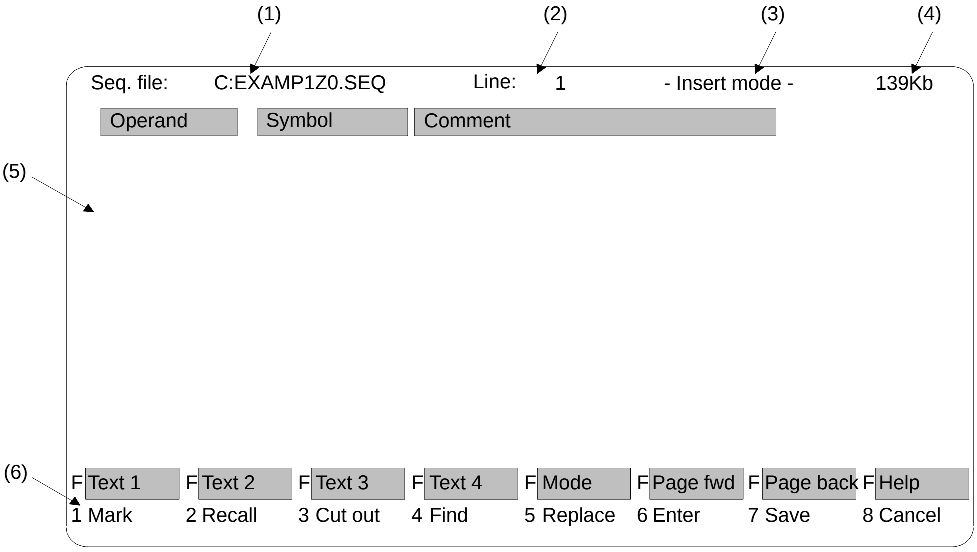
Screen layout The lines and areas of the editing field have the following significance:

Seq. File (1) Drive and name of the sequential source file (assignment list). The name is preset with the name of the symbols file selected in the settings.

Line (2) Number of the line in which the cursor is located.

Mode (3) Mode display, can be switched over between insert and overwrite mode with ***SHIFT F5*** = *Mode*.

139 Kb (4) Available or required memory space.



Seq. file: C:EXAMP1Z0.SEQ Line: 1 - Insert mode - 139Kb

Operand Symbol Comment

F Text 1 F Text 2 F Text 3 F Text 4 F Mode F Page fwd F Page back F Help

1 Mark 2 Recall 3 Cut out 4 Find 5 Replace 6 Enter 7 Save 8 Cancel

Figure 3-53 Screen Layout with Lines and Areas of the Editing Field

Editing Area

(5) This area is divided into three columns:

- Operand
  Column for entering the absolute operands.
  This column width cannot be changed.
- Symbol;
  Column for entering the symbolic operands,
  The column width depends on the setting you made in Object\Setting\Page2.
- Comment;
  Column for entering the operand comments.
  The column width depends on the setting in page 2.

Function Keys

(6) Function key menu for calling editing functions (keys marked with ”*” call further key levels). The keys have the following effects:

F1* = *Mark*	Stores a selected text (line, text field or text you have typed in) in the buffer from where you can copy the text to any part of the assignment list using ***F2**** = *Recall*. Stores texts in memory that have been typed in and can be called using the function keys ***SHIFT F1*** = *Text 1* to ***SHIFT F4*** = *Text 4*.
F2* = *Recall*	Fetches a text buffered with ***F1****= *Mark* and copies it at the cursor position.
F3* = *Cut out*	Deletes the line containing the cursor or deletes a selected passage of text. The deleted text is written to the buffer and allows text to be transferred using ***F2**** = *Copy*. A text you put in the buffer previously is lost.
F4* = *Find*	Find operands, lines, text passages or strings or go to the beginning or end of the assignment list. If you enter a search key, the text string will only be found if it is an exact match including upper and lower case letters.

F5* *= Replace*	Replaces character strings (maximum 20 characters including blanks) with another character string. The search key must be identical to the string to be replaced including upper and lower case characters.
F6 *= Enter*	Complete the editing session and store the sequential source file. The conversion to the symbols files is started automatically.
F7 *= Save*	Save the source file without conversion, e.g. if you want to take a break. You can resume work with the assignment list immediately.
F8 *= Cancel*	Cancel the editing session without storing the sequential source file.
SHIFT F1 *= Text 1*	Output text 1 with programmed function key.
SHIFT F2 *= Text 2*	Output text 2 with programmed function key.
SHIFT F3 *= Text 3*	Output text 3 with programmed function key.
SHIFT F4 *= Text 4*	Output text 4 with programmed function key.
SHIFT F5 *= Mode*	Select the editing mode: insert or overwrite.
SHIFT F6 *= Page fwd*	Page one screen down.
SHIFT F7 *= Page back*	Page one screen up.
SHIFT F8 *= Help*	Display the function key assignment.

Creating the Assignment List

1. Type in the character string for the absolute operand, e.g. ***I 1.1***.
2. Position the cursor in the symbols column using the mouse or ***TAB***.
3. Type in the character string for the symbol without preceding it with a hyphen, e.g. SIGNAL 1.

In the assignment list itself, you do **not** enter the hyphen before the symbolic operand. The column width corresponds to the symbol length you selected in Object\Settings\page1. If you do not make an entry in the symbols column (the symbols field is empty) STEP 5 displays the prompt:

”Accept absolute operand as symbol?”

Yes	The character string of the absolute operand is used as the symbolic operand in the symbols file. In the sequential source file, this field remains empty. The symbolic operand is only entered in the sequential source file following a conversion (→ *Management*, → INI > SEQ).
No	The absolute operand is not used as the symbolic operand, the field remains empty.

Operand Comments

If you want to add an explanatory text to the symbolic operands, a maximum 40 character wide comment column is available. The operand comment can also be input if you have selected ”comments: no” in the settings (→ *Project*). The operand comments (upper and/or lower case letters) are not separated, but are also stored in the symbols file.

1. Position the cursor in the comments column with the mouse or ***TAB.***
2. Type in the character string for the operand comment, e.g. ***example of a comment***.
3. Exit the line with the mouse or press the ***Return*** key.

Form Feed

If your assignment list is long, you can divide it into pages by entering a control character. To do this,

type in ”***.PA***” in the ”operand” field beginning in the first column.

You cannot make any further entries in this line.
When you call up and output the sequential source file, this control character produces a form feed in the printout. The control character is not entered in the symbols file (*Z0.INI).

Complete Editing

1. *Press* **F6** = *Enter.*

The sequential source file is stored and translated into the symbols file. If no errors occur, STEP 5 displays the message ”*n* lines processed, no errors found”.
(*n* = number of lines).

2. *Click on **OK** or press the **Return** key.*

STEP 5 exits the editor and returns to the menu.

Errors when Editing

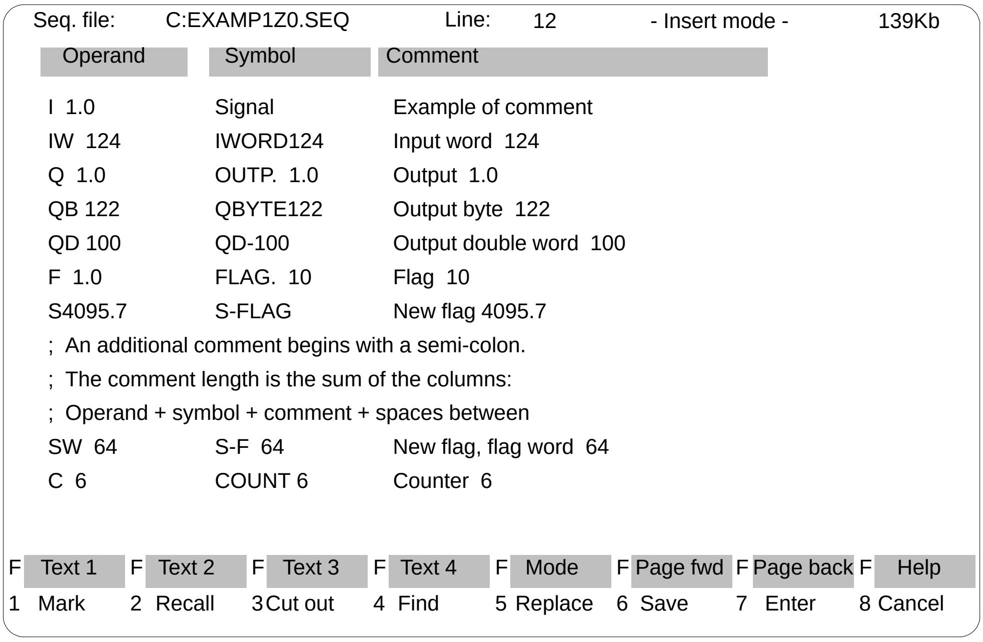
If one error occurs during the translation, STEP 5 displays the message ”error found in line n. Absolute parameter does not match OPID”. (OPID = operand identifier).The editor remains active, the incorrect line is displayed as the first line on the screen. After you have eliminated the error in the sequential source file, you can start a new translation by storing.

If several errors occur, STEP 5 displays the message ”*n* lines processed, *n* errors found”. ”Display error list?”:

Yes:	the error list is displayed
No:	you exit the editor

STEP 5 records the error in the *ZF.INI file.

You can output this error list with the management function → *Assignment lists, Output error list*.



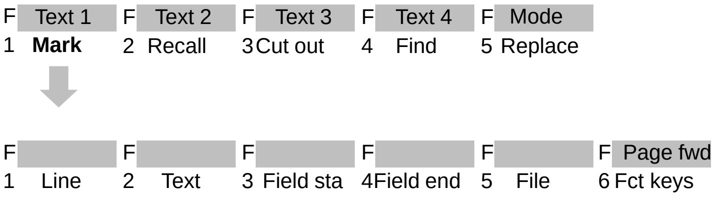
```
Seq. file:    C:EXAMP1Z0.SEQ        Line:   12          - Insert mode -          139Kb
 Operand          Symbol           Comment

 I  1.0           Signal           Example of comment
 IW  124          IWORD124         Input word  124
 Q  1.0           OUTP.  1.0       Output  1.0
 QB 122           QBYTE122         Output byte  122
 QD 100           QD-100           Output double word  100
 F  1.0           FLAG.  10        Flag  10
 S4095.7          S-FLAG           New flag 4095.7
 ;  An additional comment begins with a semi-colon.
 ;  The comment length is the sum of the columns:
 ;  Operand + symbol + comment + spaces between
 SW  64           S-F  64          New flag, flag word  64
 C  6             COUNT 6          Counter  6
```

Figure 3-54 Example of the Assignment List in the Sequential Source File

Editing Support

STEP 5 provides editing functions when you create the assignment list and they can be activated using the softkey menu. The individual functions are described below.

F1 = *Mark*



With this key, you can write selected lines, character strings and whole fields of lines to a buffer, from where you can fetch it again when it is required (copy). You can also transfer text fields to a different sequential source file.

F1
= Line Buffer the line containing the cursor so that it can be copied elsewhere.

F2
= Text Buffer a text you have typed in (max. 40 characters) for copying.

F3
= Field sta Mark the start of a field of lines (including the line in which the cursor is located).

Note on the repetition factor

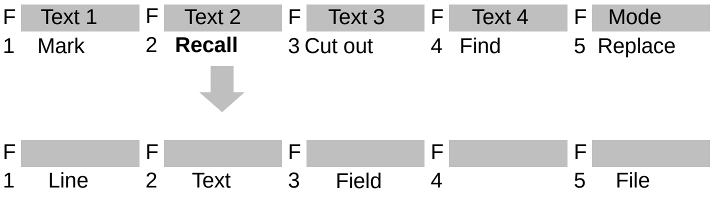
The field start character @ is set until the field is marked.

F4
= Field end Mark the end of a field of lines (including the line in which the cursor is located).
This field can also be transferred to another sequential source file, → ***F5*** *= File*

F5
= File Store the marked field in a different sequential source file. This file does not need to exist first.

F6
= Fct keys You can assign texts you have typed in (max. 40 characters) to four function keys so that you can call up regularly recurring strings during the editing session (→ *Programmable function keys*).

F2 *= Recall*



A line, text you have typed in or a field of lines is inserted before the line in which the cursor is located, i.e. copied from the buffer. You can specify a repetition factor if you wish to copy the content of the buffer several times. You can also insert a different sequential source file in the assignment list you are working on.

Note on the repetition factor

The cursor cannot be positioned on the input field for the repetition factor, it only jumps to this field after a number has been entered in the repetition factor line.

F1
= Line The marked line or a line written to the buffer with the delete function is inserted before the line in which the cursor is located.

F2
= Text The text you have typed in and marked is inserted before the line in which the cursor is located.

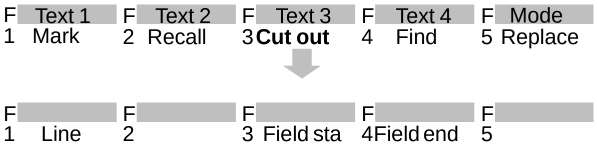
F3
= Field The marked field of lines or a previously deleted field is inserted before the line marked by the cursor.

F5
= File The marked field of lines is transferred (copied) to a different sequential source file whose name you must specify. The file must already exist, and its previous contents will be overwritten.

Note

If you accidentally overwrite a file, you can recreate the original sequential source file by generating the source file from the symbols file using → *Management, Convert INI > SEQ*. The conversion, however, ignores comments and control characters.

F3 *= Cut out*



With this function you can delete a line or field. The deleted line or field is written to the buffer. If you have already buffered a field or line this is overwritten. You can then copy the content of the buffer elsewhere → ***F2*** *= Recall*.

F1 *= Line* Delete the line containing the cursor. The line is written to the buffer.

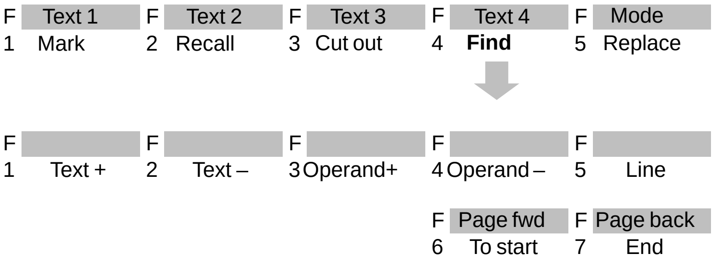
F3 *= Field sta* Mark the start of a field.

Note

The field start character @ is set until the field is marked.

F4 *= Field end* Mark the end of a field. As soon as you press this key or click on it with the mouse, the block is deleted and written to the buffer.

F4 *= Find*



The cursor is moved to a specified line or to the beginning or end of the text. It is also possible to search for operands or text strings.

F1 *= Text +* Search for a character string in the operand comments or the additional comments (following ”;”) starting from the cursor position.

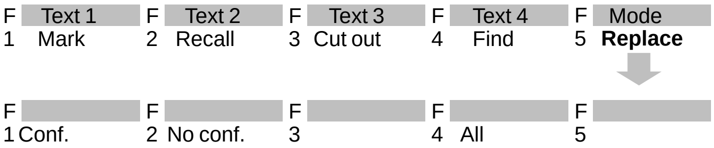
F2 *= Text –* Search for a character string in the operand comments or the additional comments (following ”;”) backwards from the cursor position.

Note

The search key must be identical to the text including upper and lower case letters.

F3 *= Operand +*	Search for absolute operands from the cursor position.
F4 *= Operand –*	Search for absolute operands backwards from the cursor position.
F5 *= Line*	Jump to the line with the specified line number.
F6 *= To start*	Position the cursor at the beginning of the file.
F7 *= End*	Position the cursor at the end of the assignment list.

F5 *= Replace*



You can replace a character string (max. 20 characters) either automatically or after a prompt for confirmation.

F1
= Conf

The character string is searched for in the assignment list n times (n = repetition factor) from the cursor position and is replaced by the new string you entered. Before it replaces a text, STEP 5 prompts you for confirmation.

Yes The characters are replaced.

No The characters are not replaced, the cursor jumps to the next character string (if n > 1) and the prompt is repeated.

Cancel: The function is stopped.

F2
= No conf
The character string is searched for in the assignment list n times (n = repetition factor) from the cursor position and replaced by the text you have typed in. No confirmation is prompted.

F4
= All
The character string is searched for throughout the whole assignment list and replaced by the new string.

Programmable Function Keys

You can assign character strings (max. 40 characters) to four function keys, so that you can insert regularly recurring text strings at any position in the assignment list. The key assignment is stored in the file *ZT.SEQ,

Programming

You have selected ”symbols: yes” in the setting (→ *Project*).

1. Press ***F1*** *= Mark.*

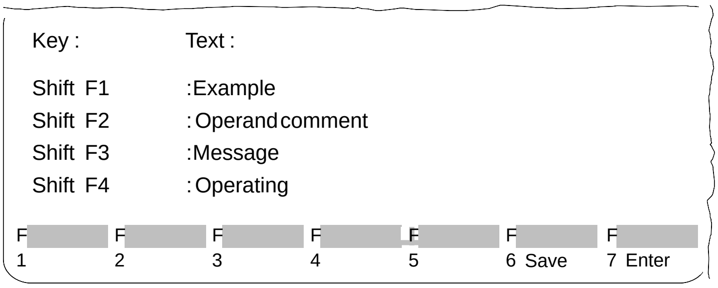
STEP 5 displays the next key level.

2. Press ***F6*** *= Fct keys.*

The editor for the function keys is displayed. The cursor is flashing in the first line.

3. Type in the character string and press the ***Return*** key.
4. Move the cursor from line to line using the ***Return*** key or ***cursor up/down*** keys.

The mouse cannot be used except to activate ***F7*** *= Enter.*



Key :	Text :
Shift F1	:Example
Shift F2	: Operand comment
Shift F3	:Message
Shift F4	: Operating

5. You can delete characters marked by the cursor using the ***DEL*** key and characters left of the cursor with ***backspace***.

To complete editing:

6. Press ***Insert*** or cancel with ***ESC***.

Modifying the Assignment List

If you want to modify an assignment list you have already created and translated, you can edit the sequential source file providing it still exists. If the sequential source file does not exist, this is generated automatically from the symbols file and output.

Please note that when editing, you cannot exceed the preset operand comment and symbolic operand length. If you want to use longer operand symbols and comment texts in an existing assignment list, you must set up a new sequential source file (→ *Project*, *Settings*) and copy the existing assignment list to this new file using the editing functions ***F2*** = *Recall* and ***F5*** = *File*.

How to Modify and Change the Field Lengths

1. Type in the ”drive” and ”name” of the new symbols file you want to create in the settings (page 1) and set the ”symbols and comment length” on page 2. These lengths must be the same or longer than the existing lengths.
2. Call the assignment list editor *(→ Editor, Assignment list)*

STEP 5 displays a new, empty assignment list.

3. Copy the file you want to change into the current file by pressing ***F2*** = *Recall* and ***F5*** = *File.*

STEP 5 displays the message: ”file name Z0.SEQ”

4. Here, enter the ”drive” and ”file name” of the existing assignment list and complete your input with the ***Return*** key.

After copying the file you can change to the editing mode with ***F8*** = *Return*. You can now edit the sequential source file as usual. To overwrite entries, change to the overwrite mode ***F5*** = *Mode*.

Inserting Lines

You can insert lines at any point. In the input mode, pressing the ***Return*** key creates an empty line below the line containing the cursor. The ***vertical expand*** key inserts an empty line above the line containing the cursor. In the overwrite mode, position the cursor at the beginning of the next line with the ***Return*** key.

Overwriting Files

When storing the modified assignment list, the existing symbols file and the sequential source file with the same name are overwritten without prompting you to confirm your intention.

Additional Comment

If there is no space for your comment, you can also add an additional comment. To do this, type in the character ”;” as the first character in the operand column followed by the required additional comment. The character ”;” marks the line as an additional comment line. The semicolon must always be in the first column of the operand field. You can enter additional comments in any line.

The number of characters available for entering an additional comment is the total of the operand length (10 characters) the preset symbol and comment length and the characters available in between the columns. Depending on the preset symbol and comment lengths, 19 to 76 characters are possible.

The special character ”;” (Fig. 3-54) can no longer be deleted by the editor. If you want to eliminate this character, you must delete the whole line (→ ***F3*** *= Cut out,* ***F1*** *= Line*).

Note

Additional comments and printer control characters only exist in the sequential source file. If you generate a sequential source file from the symbols file using → *Management, Convert INI > SEQ*, additional comments and printer control characters (.PA) are lost.

3.4 Test

This submenu includes test, information and start-up functions that you can execute on the PG in the online mode.

To use the online functions, there must be a physical and logical connection between the PG and PLC. Apart from establishing the cable connection, you must also set the correct bus path for a bus link (SINEC H1, SINEC L2 or AS511) and the mode on the PG.

The following test functions

- signal status display of operands (→ *Status variable*)
- forcing output process interface modules (→ *Force outputs*) and
- modifying process variables (→ *Force variables*)

require the listing of process variables which you can store in a variables block (VBnn (1 <= nn <= 255)) after editing. If you use the variables block, you do not have to input the operands again when you call a test function a second time. Variables blocks are stored in the program file.

Online Functions PG –PLC

Online Function	**PLC Status**	**Processing in PLC**	**Explanation**
Status block	RUN	User checkpoint	test sequence of statements in the user program
Status variable[1]	RUN	System checkpoint	output signal states of process variable (I, Q, F, S, T, C, D)
Start PLC	STOP > RUN	Start cycle	as with manual operation
Stop PLC	RUN > STOP	Stop cycle	as with manual operation
Compress memory	RUN, STOP	PLC RAM area	compress memory
Force variable[1]	RUN	System checkpoint	modify process variable (I, Q, F, S, T, C, D)
Force outputs[1]	STOP	System checkpoint peripherals	set outputs to signal state (QB, QW, QD)

Online Function	PLC Status	Processing in PLC	Explanation
ISTACK / BSTACK	STOP	PLC memory system area	output interrupt stack / block stack
Output memory contents	RUN, STOP	RAM/EPROM, S5 bus, I/Os	output memory and I/O addresses in hexadecimal
Memory configuration	RUN, STOP	PLC RAM, EPROM	data about user memory of the PLC (RAM/EPROM)
System parameters	RUN, STOP	Release of PLC SW, CPU	info about internal PLC structure and software release (CPU)
Program test ON	PROG TEST	User checkpoint	test single program steps: PB, FB, FX, OB, SB, search
Program test OFF	PROG TEST> STOP	User checkpoint	terminate program test; executed immediately

[1] Lists of operands can be stored in variables blocks (VB).

3.4.1 Block Status

Test
Block status

With this function you can test and correct blocks loaded in the PLC (user memory).

STEP 5 outputs the current signal status of the following process variables:

- inputs (I), timers (T) and counters (C)
- outputs (Q) (parameter type Q the identifier of an FB (FX))
- flags (F, S)
- data (D) (the data depends on the DB open at the time of the status output).

Status processing is subject to the following restrictions:

- The status output of the current block parameters of function blocks is only possible with the S5-135U, S5-155U and S5-115U.

- With parameter declarations (formal parameters) and the statement LIR in an FB or FX, no signal status is displayed.
- The operation DO DW/DO FW is processed along with the next operation as if it were a single operation. For this reason, only the status of the next operation is displayed.
- Some operations terminate the status processing mode, since following their execution a branch is made to the operating system or to other blocks, e.g. LIR, BEC and all jumps and blocks calls.
- A hardcopy is always possible after status processing has been terminated.
- While status processing is active, the mouse cannot be used.

After you select the function in the test menu, the selection box “Block status” appears. Here, you specify the block to be tested (→ *Graphical user interface, Selection box*).

Input field	Explanation
Block	– **Without nesting:** type in the block type and block number or symbolic name of the block. – **With nesting:** type in the block to be tested first and then the sequence of blocks preceding it in the program (maximum 5) via which the block status is to be displayed during the test.
Search Key	Here, you can specify the search key of the statement to be tested. STEP 5 automatically searches for this and displays the block section containing this term on the screen. All possible search keys are listed in the help box.
Overwrite	In this window, you specify whether STEP 5 overwrites the old block directly following modifications or only after user confirmation.
Seq. source file	Here, you must enter an “X” to specify whether STEP 5 updates the Z0.SEQ file or not.

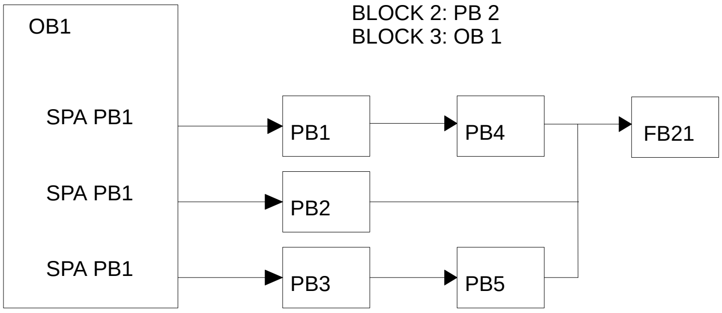
Example of Nesting

You want to display the status of FB 21 when this has been called by PB 2. In this case, you enter the blocks as follows:

FB 21, PB 2, OB 1

Nesting of the blocks:

Block list:
BLOCK 1: FB 21
BLOCK 2: PB 2
BLOCK 3: OB 1



Representation of the Signal Statuses on the Screen

STL: The signal states are displayed as a list of status information.

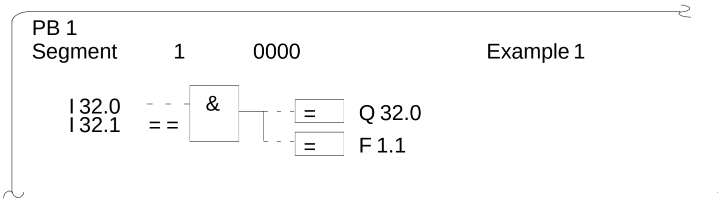
LAD/CSF: In the Ladder Diagram and Control System Flowchart, the signal states are indicated by the way in which the connection lines are displayed.

========= Signal state 1

. . . . Signal state 0

– – – – – – – – – – Signal state cannot be represented (for example, is not one of the 20 displayable statements; the number of statements depends on the PLC).

After **OK**, STEP 5 begins the status processing and displays, e.g. the following screen in CSF:



```
PB 1
Segment        1          0000                        Example 1
```

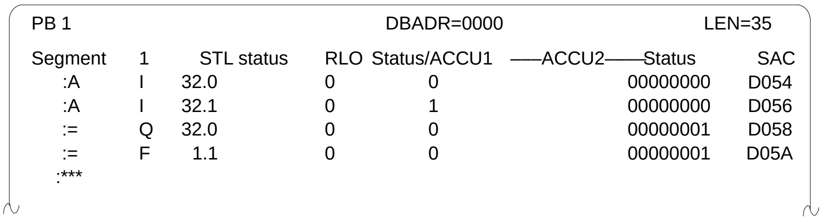
Figure 3-55 Status processing

The display is **not** updated following each cycle.
All the functions made available in the function key menu (→ *Editor, STEP 5 blocks*) can be executed during status processing.

Note

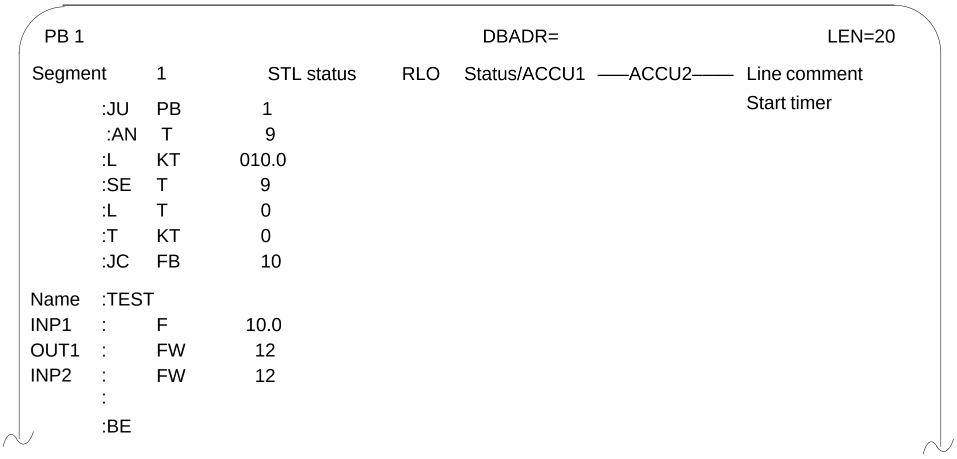
You cannot display addresses.

In STL, STEP 5 displays the following screen (example):



```
PB 1                                     DBADR=0000                    LEN=35

Segment    1     STL status      RLO  Status/ACCU1   ——ACCU2——Status      SAC
    :A     I    32.0             0         0                 00000000    D054
    :A     I    32.1             0         1                 00000000    D056
    :=     Q    32.0             0         0                 00000001    D058
    :=     F     1.1             0         0                 00000001    D05A
   :***
```

Figure 3-56 Screen Layout



```
PB 1                                              DBADR=                                   LEN=20

Segment      1           STL status     RLO   Status/ACCU1  ——ACCU2——  Line comment
                                                                        Start timer
        :JU    PB         1
         :AN    T          9
        :L     KT       010.0
        :SE    T          9
        :L     T          0
        :T     KT         0
        :JC    FB         10

Name    :TEST
INP1    :      F        10.0
OUT1    :      FW        12
INP2    :      FW        12
        :
        :BE
```

The display is **not** updated after each cycle.

Abbreviations

RLO	Result of logic operation
STATUS	Bit operands
DBy	Current data block
ACCU 1	Content of ACCU 1
ACCU 2	Content of ACCU 2
STATUS	Status of the result condition code bits
SAC	Step address counter

Identifiers for status display:

R	Timer running
N	Negating bit scan, i.e. with the AT (AND timer) the result is 0
U	Upwards counter input
D	Downwards counter input
S	Set and start input
E	Enable input

All the functions made available with the function keys (→ *Editor, STEP 5 blocks*) can be executed during status processing with the exception of displaying addresses.

Block status processing

Action	**Operation**	**Messages/Explanations**
Move breakpoint	*Move the cursor before the required operand with cursor keys or search function. Fetch other segments onto the screen with cursor keys or "+"/"–".*	STEP 5 continues status processing. Message: "Status processing active".
Abandon processing	*Press **ESC** (cancel) once.*	The message "Status processing active" is cleared.
Continue processing	*Press **INSERT** (enter) once.*	Message: "Status processing active".
Correct program	*Press **F6-Edit**.* *Same operations as in the editor mode.*	Status processing is stopped and you change to the editor mode.
Enter correction	1. *Press **INSERT**(Enter)* 2. *Acknowledge with **yes**.* 3. *Acknowledge with yes if you want to "overwrite".*	Prompt "Enter modified segment?" "...already in PLC, overwrite?" The corrected block is in the PLC and status processing is restarted.
Stop/terminate processing	1. *Press **ESC** (cancel) twice.* 2. *Confirm prompt with **yes**.*	Prompt "Exit status?

Messages

Possible messages:	Causes:
"Statement not processed"	• block is not called • statement is skipped • a block or sequence of blocks does not exist • the PLC is in the STOP mode
"Block does not exist in PLC"	• the block to be tested does not exist • the block to be tested calls a further block that does not exist in the PLC.

3.4.2 Status Variable

Test

Status variable

Using this function you can output the current signal statuses of selected operands in the form of a list as they occur at the system checkpoint (→ *Appendix A2, Glossary*) during program execution. You enter the operands to be monitored (process variables) in a list which STEP 5 displays as an empty table when you call the “status variable” test function, providing no variables are entered otherwise the last table saved is displayed (variables block). With ***F6*** *= Activate*, or with the ***Insert*** key you can display the current signal state of the listed operands.

The listed operands are called during status processing and their current signal status is displayed before they are modified by the user program.

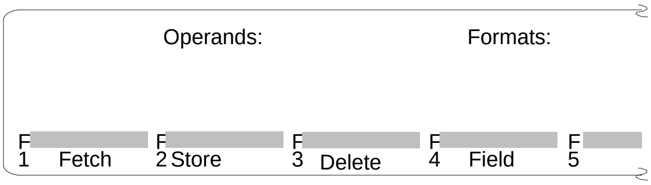


Figure 3-57 Table for Editing the Operand List

The following functions are available for this table:

Key:	Function
F1 *= Fetch*	*Call a variables block*
F2 *= Store*	*Store the operand list as a variables block*
F3 *= Delete*	*Delete the current line*
F4 *= Field*	*Display variables in fields, keys + or – fetch the previous or next field.*
F6 *= Activate*	*Activate status processing (= enter). Only available when at least one operand is entered*
F7 *= Return*	*Saves the operand list in the current variables block (only available when at least one operand is entered)*
F8 *= Return*	*Return to menu selection*
SHIFT F8 *= Help*	*Information about certain activities*

Safety prompt

If you make changes when entering the operand list, that are not saved in the variables block, you will be asked whether you want to keep the changes. Answer with <Yes> or <No>:

- Cancel (ESC)
- F8 = Return
- F1 = Fetch

The text of the prompt depends on whether or not you have selected a variables block.

No variables block selected:	Discard changes?
Variables block selected:	Discard modified block?

Action	Reaction to <Yes>	Reaction to <No>
Cancel **F8 Return**	Changes are discarded; STEP 5 displays the function menu.	You remain editing the operand list,changes can be saved in a variables block. **Note:** The changes must be explicitly saved (F2 Store or F7 Save).
F1 Fetch	Changes are discarded; After you complete the command line, specify the variables block VBnn.	You remain editing the operand list,changes can be saved in a variables block. **Note:** The changes must be explicitly saved (F2 Store or F7 Save). Call a new variables block with F1 Fetch.

Editing the operand list

You can enter the following operands in the operand list:

Operand	Permitted data formats	
F/Q/I/S	KM	
FY/QB/IB/SY	KH	(KM, KY, KS, KF)
FW/QW/IW/SW	KH	(KM, KY, KS, KF)
T	KT	(KM, KH)
C	KC	(KM, KY, KS, KF)
DW/DL/DR	KH	(KM, KY, KS, KF)
DB	–	
FD/QD/ID/DD/SD	KH	(KG, KY, KS)

After you type in an operand, the PG displays the first format, i.e. the format not in brackets, in the table above. You can overwrite this format when making your input.
With the operands DD, DW, DB, DL, DR, you must first specify the corresponding data block in the operand list. Otherwise, the PG displays the message “No DB selected”.

You must type in the characters of an operand in the correct order (syntax) otherwise the cursor remains in the input field.

You can save the operand list in a **variables block** (VB). Call an existing variables block with ***F1*** = ***Fetch***.

Note

The last variables block (VB) you saved is loaded automatically when you call “status variable”.

Editing Operations

Function	Operation	Messages/Explanations
Input an operand	1. *After you input the operand press **double arrow key right*** 2. *Change or keep format* 3. *Complete line with **Return***	The PG suggests a data format for each operand. The cursor is positioned on the operand. The cursor jumps to the beginning of the next line.
Correct	*Overwrite incorrect input*	If the syntax is wrong the cursor only leaves the field after it has been corrected.
Insert operand	*1. Position cursor with **cursor key** (up/down)* *2. Press **expand vertically*** *3. Type in operand*	
Add operand at start	1. *Position cursor in top line* 2. *Press **expand vertically*** 3. *Type in operand*	You can append operands to the list when the cursor is positioned below the last line of the list.

Function	Operation	Messages/Explanations
Delete operand	1. *Position cursor on the first character of the operand* 2. *Press* ***delete character*** *several times*	
Delete line	1. *Position cursor on the line to be deleted* 2. *Press* ***F3*** *(Delete)*	The current line is deleted with the operand and format, the next lines are closed up.
Fetch operand list	1. *Press* ***F1*** *(Fetch)* 2. *Complete the command line* Display variables block VBnn	If you have made changes, that were not saved in a variables block, a prompt is displayed ("Discard changes?" or Discard modified block?") If you did not make changes or if you answer the prompt with ***Yes,*** STEP 5 fetches the operand list from variables block VBnn after you have completed the command line.
Save operand list	*Press* ***F7*** *(Save)*	STEP 5 saves the operand list in the currently selected variables block. In contrast to ***F2*** (Store), you do not specify a variables block number. The function is only available when a variables block is selected.
Store operand list	1. *Press* ***F2*** *(Store)* 2. *Fill in the command line* Store variables block VBnn	STEP 5 stores the operand list in the variables block VBnn.
Fetch operand list as field	1. *Press* ***F4*** *(Field)* 2. *Fill in the command line* Field display from variable: QB 26 Format: KH	STEP 5 displays an operand list with 20 consecutive bytes starting from output 26.

The operand list can contain a maximum of 20 operands (if you are using words, this reduces to 10 and for double words 5).

At the bottom edge of the screen you can see what percentage of the operand list is already completed.

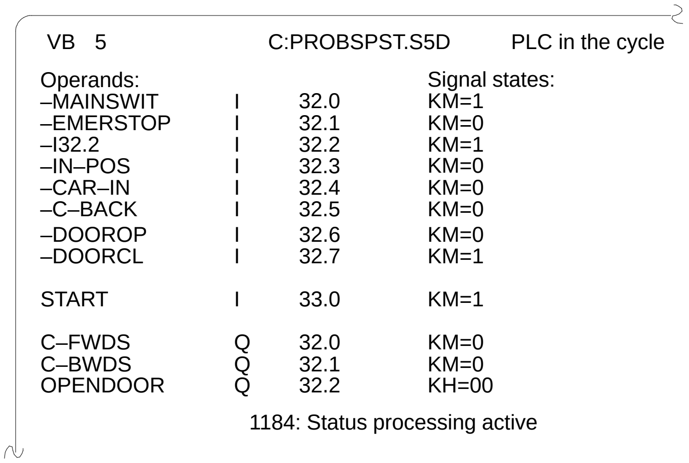
3.4

Status of the Operands (outputting process variables)

The current signal statuses of the process variables in the operand list are output before you modify the user program (i.e. at the system checkpoint).
Once you have edited the operand list or have displayed it on the screen,

press ***F6*** = *Activate* or ***Insert.***

The PG displays the signal statuses of the listed variables and the message “status processing active”.



```
 VB   5                C:PROBSPST.S5D       PLC in the cycle

Operands:                              Signal states:
–MAINSWIT        I      32.0          KM=1
–EMERSTOP        I      32.1          KM=0
–I32.2           I      32.2          KM=1
–IN–POS          I      32.3          KM=0
–CAR–IN          I      32.4          KM=0
–C–BACK          I      32.5          KM=0
–DOOROP          I      32.6          KM=0
–DOORCL          I      32.7          KM=1

START            I      33.0          KM=1

C–FWDS           Q      32.0          KM=0
C–BWDS           Q      32.1          KM=0
OPENDOOR         Q      32.2          KH=00

                 1184: Status processing active
```

Figure 3-58 Operand List with Binary Inputs/Outputs and a Flag Byte

Operation During Status Processing

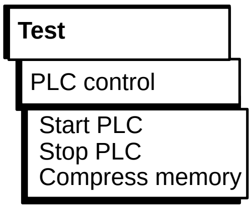
Action	Operation	Messages/Explanation
Interrupt status processing	*Press **ESC***	The cursor jumps to the first line in the operand list.
Continue status processing	*Press **F6** = Activate*	STEP 5 displays the status of the individual variables again.
Terminate/abortstatus processing	*Press **ESC** twice*	If you have made changes, that were not saved in a variables block, a prompt is displayed (“Discard changes?” or Discard modified block?”) If you did not make changes or if you answer the prompt with ***Yes,*** STEP 5 displays the function menu.

Possible Messages and Operator Errors

Messages	Causes
”No DB selected”	You have not specified the data block for an operand.
”KH= *data element missing”	The DB for the specified operands (DD, DW, DB, DL, DR) is not in the PLC memory or there are not enough data words.
”KT = stopped”	The selected timer was not started.
”KH = * DB missing”	The DB does not exist in the selected program file.
”* illegal”	Operand not allowed in the PLC

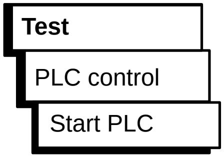
3.4.3 PLC Control

PLC Control



Within this submenu, you can start and stop a PLC connected online and compress the user memory in the PLC.

Starting the PLC



The function “start PLC” triggers a cold restart or warm restart on the programmable controller (refer to your PLC manual).

Before the PLC is started with this function, you are prompted by the PLC to confirm your intention.

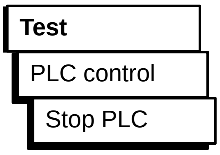
– Acknowledge the message with ***yes:***

The PLC is set to the selected status,
or

– Acknowledge the message with ***no:***

The PLC is not started.

Stopping the PLC



The function “Stop PLC” switches the programmable controller to the STOP mode (refer to your PLC manual). The processor stops executing program statements.

In multiprocessor operation (S5-135U) all the processors are set to the stop mode.

Before the PLC is stopped with this function, you are prompted to confirm your intention.

– Acknowledge the message with ***yes:***

The PLC is set to the stop mode,
or

– Acknowledge the message with ***no:***

The PLC does not stop.

Compressing the PLC Memory

Test
PLC control
Compress memory

When you delete blocks in the PLC, these are declared “invalid” in the PLC RAM but are not physically deleted. Whenever you correct a block, the old version of the block is invalidated but remains in memory and the corrected block is written into the RAM. This means that the PLC memory can become full. The “compress memory” function deletes invalid blocks and shifts valid blocks together so that there is memory again for new blocks.

The “compress memory” function detects the following errors:

- wrong block length,
- corrupted pattern “7070” in the block header,
- invalid block type (with OB, invalid block number).

If STEP 5 detects one of these errors, the function is abandoned and a corresponding message is displayed.

3.4.4 Force Variables

Test
Force variables

This online function allows you to modify process variables and to intervene directly in the process. Before you force variables, you should consider the reaction of the process to your intervention!

- The variables I, Q, F, S, T, C, D can be modified. The PG influences the variables I, Q and F only in bytes or words in the process image.
  With the variables T and C in the format KM and KH, the forcing of edge flags must be taken into account.
- The function can be executed in the STOP and RUN modes of the programmable controller.
- The signal status display is stopped if an incorrect format or operand is found.
- STEP 5 displays the message “forcing not possible”.
- Since STEP 5 modifications are made in bytes, variables cannot be modified *en bloc*.

How to use "Force Variables"

The following procedure is advisable when working with the "force variables" function:

1. Call "force variables".
   STEP 5 displays an empty table for the operand list providing no variable is entered. Otherwise the last variables block you saved is displayed.
2. Make your entries in the operand list and complete the editing with the ***Insert*** key.
   The status of the variables is displayed.
3. Cancel the status display with ***ESC***.
   The operand list with the current values is displayed.
4. Modify the current values and complete your input with the ***Insert*** key.

From point 2 onwards you can repeat the procedure.

Operation

After selecting the "force variables" function, STEP 5 displays the empty table for entering the operand list (Fig. 3-58) or the variables block last selected for "force variables".

Editing the Operand List

You can enter the following operands in the operand list:

Operand	Permitted data formats	
F/Q/I/S	KM	
FY/QB/IB/SY	KH	(KM, KY, KS, KF)
FW/QW/IW/SW	KH	(KM, KY, KS, KF)
T	KT	(KM, KH)
C	KC[1]	(KM, KY, KS, KF)
DW/DL[1]/DR[1]	KH	(KM, KY, KS, KF)
DB	–	
FD/QD/ID/DD/SD	KH	(KG, KY, KS)
–Symbol	Dependent on operand type	

[1] These operands and formats can only be seen (not controlled).

After you type in a byte or word operand, STEP 5 displays the first format, i.e. the format not in brackets, in the table above. You can overwrite this format when making your input.
With the operands DD, DW, DB, DL, DR, you must first specify the corresponding data block in the operand list. Otherwise, STEP 5 displays the message “No DB selected”.

You must type in the characters of an operand in the correct order (syntax) otherwise the cursor remains in the input field.

You can store the operand list in a **variables block** (VB). Call an existing variables block with ***F1*** = *Fetch*.

The operand list can contain a maximum of 20 operands (with word operands, the maximum is reduced to 10, and with double words 5). At the lower edge of the screen, the occupation of the operand list is displayed as a percentage.

The editing options are the same as for the “status variables” function, on page 3-218.

Note

The last variables block (VB) you saved is loaded automatically when you call “force variables”.

Status of the Operands (displaying process variables)

The current signal statuses of the process variables in the operand list are output.
Once you have edited the operand list or have displayed it on the screen,

– Press ***F6*** = *Activate* or ***Insert****.*

The PG displays the signal statuses of the listed variables and the message “status processing active”.

To interrupt status processing,

– Press ***ESC*** = *Cancel.*

The cursor jumps to the first line of the operand list.

Influencing Process Variables from the PG

The current signal state of the listed process variables is displayed on the screen. You can now modify the values of the displayed process variables in the PLC (force variables).

Modifying the Value of a Variable

The PG displays the operand list with the column “signal states” in which the currently valid signal states are displayed. The message “status processing active” and the PLC mode are also displayed.

1. Press ***ESC*** = *Cancel* once.
   The PG changes the name of the “signal statuses” column to “force process image” and waits for you to input the forced values. The cursor jumps to the first line.
2. Enter the forced values line by line and press the ***Return*** key after each input.

You complete the input of variable values by

3. Pressing the ***Insert*** key.
   STEP 5 displays the message “End of force fct.” and transfers the modified variables to the PLC.
4. Pressing the ***Insert*** key.
   The PG changes the name of the “Force” column to “Signal states”. You can see changed signal states.

To stop the force variables function,

5. Press ***ESC*** = *Cancel* twice.
   If you have made changes, that were not saved in a variables block, a prompt is displayed (“Discard changes?” or Discard modified block?”)
   If you did not make changes or if you answer the prompt with ***Yes,*** STEP 5 returns to the basic functions menu.

3.4.5 Force Outputs

Test

Force outputs

With this function you can set outputs to the required signal state directly. The function does not influence the process image or program execution, since the programmable controller must be in the STOP mode.

The outputs of a programmable controller can be forced individually. You can therefore check their assignment to the actuators of your plant (e.g. valves, motor etc.). With this function you can check whether output modules are defect or not plugged and that the wiring is correct.

Single bits cannot be addressed, but only the formats byte, word and double word.

How to use "Force Outputs"

The "force outputs" function is used as follows:

1. Change the PLC to "STOP"
2. Call the "force outputs" function.
   STEP 5 displays an empty table for the operand list providing no operand is entered. Otherwise, the variables block you saved last is displayed.
3. Enter the operands and complete the list with the ***Insert*** key.
4. Type in or modify the required values and complete your entries with the ***Insert*** key .
   The PG transfers the values to the outputs of the PLC.

From the third point onwards you can repeat the procedure.

When you select the "force outputs" function, STEP 5 displays the empty table for the operand list (Fig. 3-58) or the variables block last selected for "force variables".

Editing the Operand List

You can enter the following operands in the operand list:

Operand	Permitted formats	
OB	KH	(KM, KY, KS, KF)
OW	KH	(KM, KY, KS, KF)
OD	KH	(KG, KY, KS)
–Symbol	Dependent on operand type	

Inputting Operands

After you type in an operand, STEP 5 displays the first format, i.e. the format not in brackets, in the table above. You can overwrite this format when making your input.
You must type in the characters of an operand in the correct order (syntax) otherwise the cursor remains in the input field.

You can store the operand list in a variables block (VB). Call an existing variables block with ***F1*** *= Fetch*.

The operand list can contain a maximum of 20 operands (with word operands, the maximum is reduced to 10, and with double words 5). At the lower edge of the screen, the percentage of the operand list completed is displayed.

The formats of the operands depend on the PLC type:

S5-130 W/A, S5-150 A/K	: QB, QW;
S5-150 S, S5-135 U, S5-155 U	: QB, QW, QD;

The editing options are explained on page 3-232.

Setting Output Variables at the PG

STEP 5 displays the last selected variables block or an empty list in which you can enter signals and states.

Modifying Output Values

STEP 5 displays the operand list with the columns “Operands” and “Force I/O modules”.

1. Type in the required forced values line by line and press the ***Return*** key after each input.

STEP 5 displays an **X** after each entered value. If you type in less characters than the maximum length, the more significant places are automatically padded with zeros.

To complete the entry of input values:

2. Press the ***Insert*** key.

The PG displays the message “End of force fct.” and transfers the modified output values to the PLC.

If you want to stop the force outputs function,

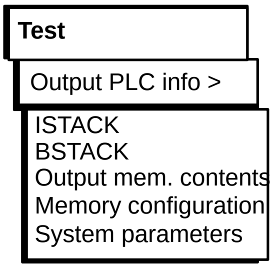
3. Press ***ESC*** *(cancel).*

If you have made changes, that were not saved in a variables block, a prompt is displayed (“Discard changes?” or Discard modified block?”).
If you did not make changes or if you answer the prompt with ***Yes,*** STEP5 returns to the basic menu of the functions. Refer to 3.4.2 Status Variable”

Corrections

The cursor only exits the input field when the input is correct. If you make errors inputting the values the cursor remains in the input field.

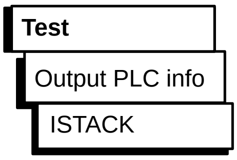
3.4.6 Outputting PLC Info



The online functions you can select in this submenu provide you with information about the status of the connected PLC.

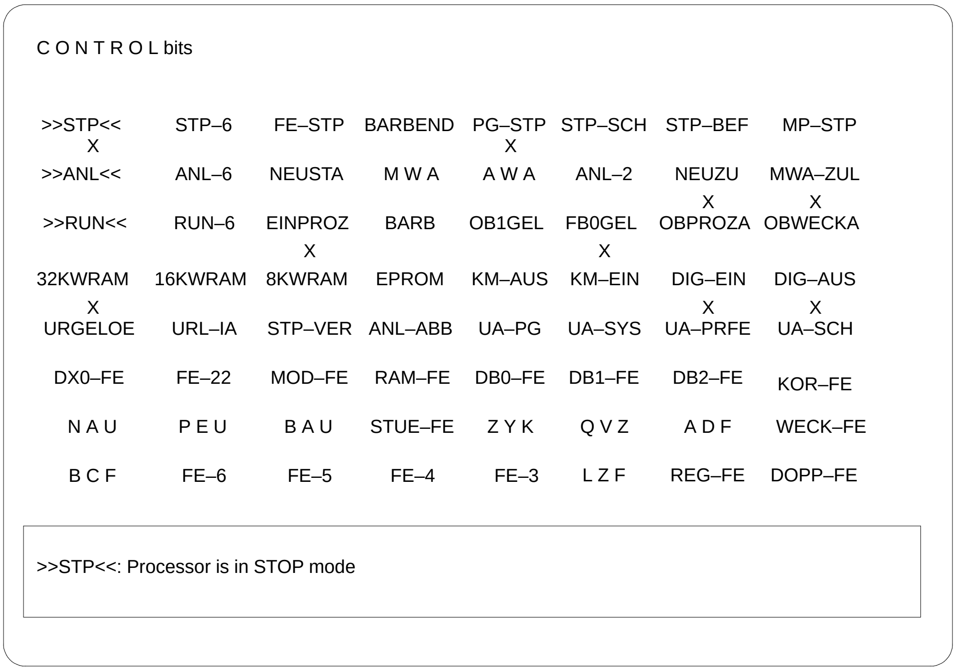
- Interrupt stack (*ISTACK*)
- Block stack (*BSTACK*)
- Memory and I/O addresses, hexadecimal (*output memory contents*)
- Information about the user memory on the PLC (*memory configuration*)
- Information about the internal PLC structure and the software releases of the CPU (*system parameters*)

ISTACK Interrupt Stack of the PLC



After you select the ISTACK, a table of control bits and their current settings is displayed on the screen. You can select the abbreviations using the cursor and an explanation of the currently marked abbreviation is displayed in a window at the lower edge of the screen.

The control bits are explained in detail in the PLC manuals.
To display the control bit screen form, the PLC does not need to be in the STOP mode.



```
C O N T R O L bits

>>STP<<    STP–6     FE–STP    BARBEND   PG–STP    STP–SCH   STP–BEF   MP–STP
   X                                        X
>>ANL<<    ANL–6     NEUSTA    M W A     A W A     ANL–2     NEUZU     MWA–ZUL
                                                             X         X
>>RUN<<    RUN–6     EINPROZ   BARB      OB1GEL    FB0GEL    OBPROZA   OBWECKA
                        X                             X
32KWRAM    16KWRAM   8KWRAM    EPROM     KM–AUS    KM–EIN    DIG–EIN   DIG–AUS
   X                                                         X         X
URGELOE    URL–IA    STP–VER   ANL–ABB   UA–PG     UA–SYS    UA–PRFE   UA–SCH

DX0–FE     FE–22     MOD–FE    RAM–FE    DB0–FE    DB1–FE    DB2–FE    KOR–FE

N A U      P E U     B A U     STUE–FE   Z Y K     Q V Z     A D F     WECK–FE

B C F      FE–6      FE–5      FE–4      FE–3      L Z F     REG–FE    DOPP–FE

>>STP<<: Processor is in STOP mode
```

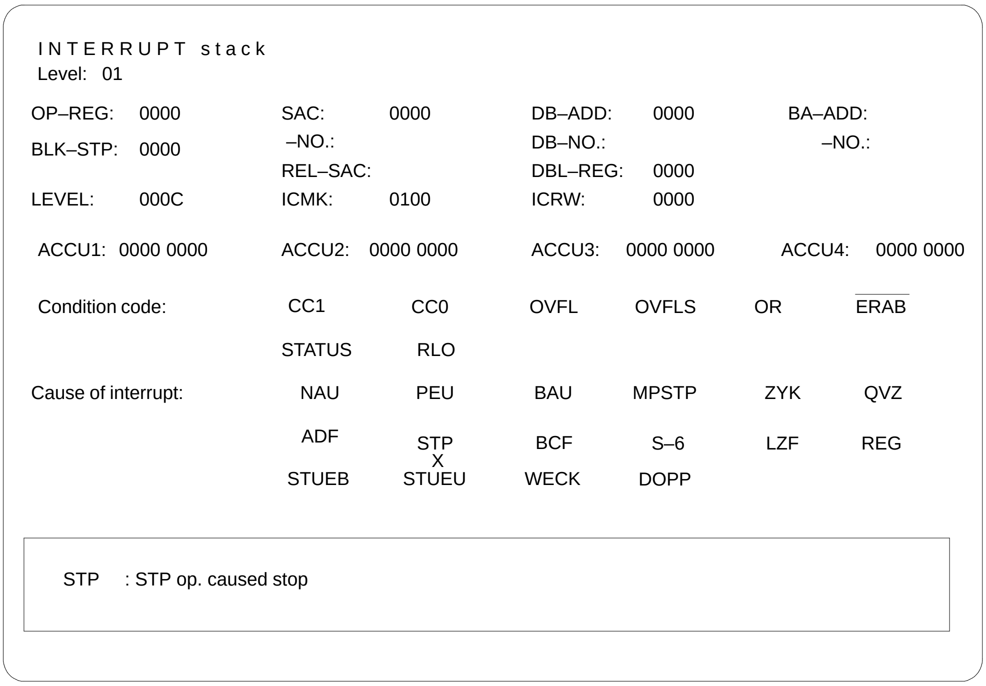
Figure 3-59 Table of Control Bits (e.g. CPU 928 B)

Once the control bit table is displayed, you can display the ISTACK by changing the PLC to the STOP mode and pressing the

1. Press the ***Insert*** key.

How you handle the plain text display is explained in a window at the lower edge of the screen which you select by pressing

2. Press ***HELP***.



```
 INTERRUPT stack
 Level:  01

OP–REG:    0000         SAC:        0000         DB–ADD:     0000          BA–ADD:
BLK–STP:   0000          –NO.:                   DB–NO.:                      –NO.:
                        REL–SAC:                 DBL–REG:    0000
LEVEL:     000C         ICMK:       0100         ICRW:       0000

 ACCU1:  0000 0000      ACCU2:  0000 0000        ACCU3:  0000 0000        ACCU4:  0000 0000

 Condition code:         CC1         CC0          OVFL       OVFLS       OR          ERAB

                        STATUS       RLO

Cause of interrupt:      NAU         PEU          BAU        MPSTP       ZYK         QVZ

                         ADF         STP          BCF        S–6         LZF         REG
                                      X
                        STUEB       STUEU         WECK       DOPP

     STP    : STP op. caused stop
```

Figure 3-60 Display of the Interrupt Stack

Note

There may be more than one screen page.

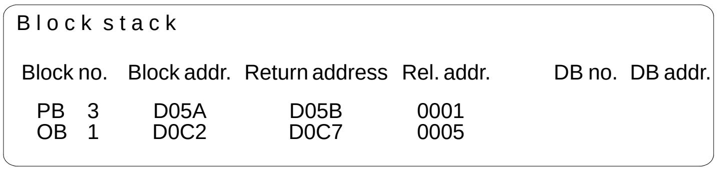
Block Stack of the PLC (BSTACK)

Test

Output PLC info

BSTACK

Each time a block is called, the PLC enters the start address of the currently valid block along with the relative and absolute return address in the block stack. The return address is the memory address at which the program must be continued once the newly called block has been processed.
You can call up this information using the BSTACK function when the PLC is in the STOP mode.



```
Block stack

Block no.   Block addr.   Return address   Rel. addr.        DB no.   DB addr.

  PB   3       D05A           D05B           0001
  OB   1       D0C2           D0C7           0005
```

Possible message:

1. *“Wrong mode at PLC”*

The PLC is not in the STOP mode.

2. *Empty or incomplete stack.*

Outputting Memory Contents

Test

Output PLC info

Output memory contents

This function outputs the absolute addresses and their contents on the screen, printer or to a print file.

The output of the addresses is only possible in the online mode.

Note

Manipulation can cause undefined statuses in the PLC – think out the consequences before you make changes.

Operation

The PG displays the job box “PLC INFO: output memory contents”.

1. Under “Output from address:” enter the first byte address to be output as a hexadecimal number (e.g. ADAC, for S5-155U (20 bit address): e.g. FADAC).
2. Press the ***Insert*** key.
   STEP 5 displays the addresses and their contents rolling the screen downwards in columns.

The address output always begins at an even address.
Non-configured memory areas are marked with **XX.** STEP 5 outputs a maximum of 1024 absolute addresses.
To freeze/interrupt the address output:

3. Press ***ESC*** = *Cancel.*
   To continue the output, confirm the prompt or press the **Insert** key.

If you want to make corrections:

4. Click on ***correction*** and position the cursor on the relevant value with ***SHIFT + cursor right/left****.*
5. Enter the value and complete your input with the ***Insert*** key.

The message “Enter modified addresses in PLC?” appears.

6. Click on ***yes*** or ***no****.*

To stop and exit the output function

7. Press ***ESC*** = *Cancel* twice.

No correction	Press ***ESC*** once and reply to the prompt with NO.
After correction	Modified addresses are output: acknowledge the message.

Memory Configuration of the PLC

Test

Output PLC info

Memory configuration

With this function, you can see the configuration and amount of user memory being used. The addresses are displayed in hexadecimal form. The memory assignments and configuration options are described in the programming instructions for the specific PLC.

On the screen, you can see the size of the user memory of the PLC and the amount currently occupied either in graphical or text form. The display differs depending on the performance of the PLC.

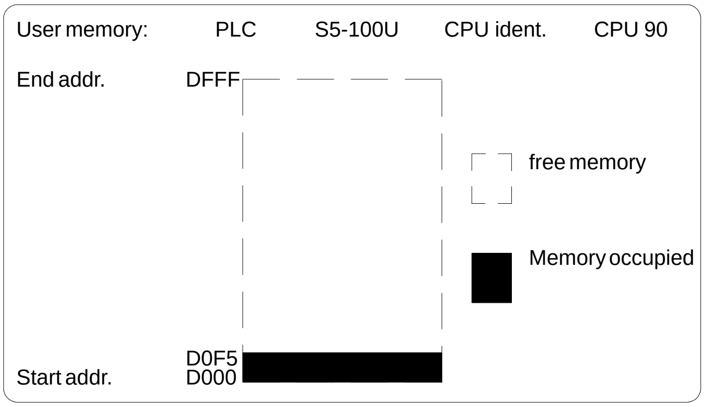
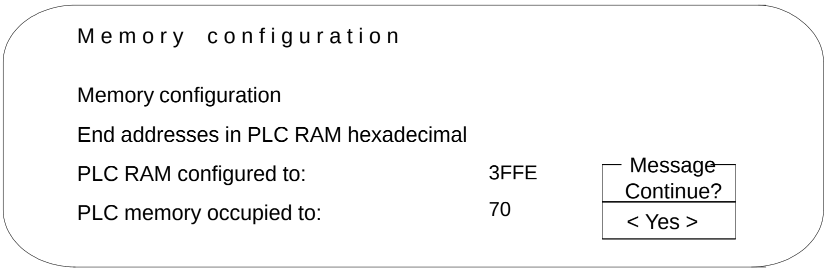


Figure 3-61 User Memory Size and Assignment on the S5-100U



Memory configuration

Memory configuration

End addresses in PLC RAM hexadecimal

PLC RAM configured to: 3FFE

PLC memory occupied to: 70

Figure 3-62 User Memory Size and Assignment as Text

System Parameters of the PLC

Test
Output PLC info
System parameters

With this function, you can display the following PLC system parameters on the screen:

- release of the PLC software
- CPU identifier
- CPU type
- CPU number
- memory distribution
- block list lengths

Operation

STEP 5 displays the PLC system parameters on the screen. The list is spread over two screen pages. The following illustration is an example of page 1. To move onto page 2 or to terminate the function, confirm the prompt “*continue*” with ***yes***.

```
System parameters

Numbers specified in hex.

PLC software release        Z 01

CPU identifier                   AG 100 U CPU 90

PGAS software release       Z 00

I/O module inputs    0
I/O module inputs    0
Process image inputs             EF00
Process image outputs            EF80
Flag memory                      EE00
Timer memory                     EC00
Counter memory                   ED00
RS memory area                   EA00
```

3.4.7 Program Test ON

Test
Program test ON

With this function, the PLC processes a block step by step. When you invoke the program test function, the program is stopped at the point marked by the breakpoint (statement in which the cursor is located) and the command output is disabled (all outputs blocked). This means that the program is only processed as far as the selected statement and the current signal states and the RLO are output. On the PLC the BASP LED is lit (block all outputs).

Note

Not all PLCs support the program test function, refer to your PLC manual.

In the program test mode

- the processing cycle is stopped,
- no inputs or outputs are processed, only the process image can be modified,
- the program can be moved on operation by operation by moving the breakpoint.

In the program test mode, the PLC stops at the last selected breakpoint. You can select the following test functions (allowing corrections to be made if necessary) parallel to the program test:

- Status variable
- Force variables
- Force outputs
- Info about the **I**nterrupt **STACK**
- Info about the **B**lock **STACK**

Special features of the program test function for specific programmable controllers are described in the PLC manuals.

After calling the “Program test ON” function, enter the following information in the box under “Selection”:

1. the block (absolute or symbolic name) or a list (nesting) of blocks you want to check.
2. as “search key”: an operand you want to check in the block you have selected.
3. Then click on ***OK***.
   STEP 5 displays the selected block in STL. The screen representation is the same as that for block status (see page 3-226 **)**. Instead of the function “Status” the “Program test” function is displayed.

4. Press the ***cursor down*** key.
   The breakpoint is selected.
   STEP 5 displays information about the operation that has just been executed. The cursor is positioned in the next statement line. The processor of the PLC is stopped, i.e. no operation in the user program is executed unless you trigger it explicitly.
5. Press the ***cursor down*** key.
   The next breakpoint is selected.
   The PLC executes the next operation and the processor stops the processing again.
   If you discover an error that needs correcting, proceed as follows:
6. Press ***ESC*** = *Cancel* twice. To exit the program test, call an editor.
   Since the program test function is still active, the processor of the PLC is stopped.

To return to the "program test" mode

7. Call the "program test ON" function again.
   You can now test the corrected program.

Note

Not all function keys are active.

Program Test OFF

Test

Program test OFF

This function deactivates the program test.

Select "Program test OFF".

The PLC changes to the STOP mode and must be restarted (→ *PLC control*, *Start PLC*) or by changing the CPU selector from STOP to RUN).

3.5 Management

This main menu includes a series of utilities which you require when working with the STEP 5 editing and test functions.

3.5.1 Generate XRF

Generate XRF

Management

Generate XRF

With this function you can generate a reference list (cross reference list) of the default program file in a file with the name *XR.INI. This is the source for cross references in LAD, CSF and STL segments in the I/Q/F list, in the program structure and in checklists and for the printout of the cross reference list itself. If you make corrections in a STEP 5 program, you must regenerate the reference list.

After triggering the function in the main menu, this function is executed automatically.

The generated reference list is required in the block editor for documentation in the KOMDOK format and in GRAPH 5 for processing the ***F2*** functions = *Reference*.

XRF files (cross-reference lists) can also be generated in the block editor and before KOMDOK output.

3.5

3.5.2 EPROM

Management

EPROMs

With this function, you transfer STEP 5 blocks from a program file to an EPROM/EEPROM submodule. This is commonly known as “blowing” an EPROM.

These memory submodules must be inserted in an EPROM port on the PG. You should only insert the module in a slot on the programmable controller (PLC) after you have transferred the block to the submodule.

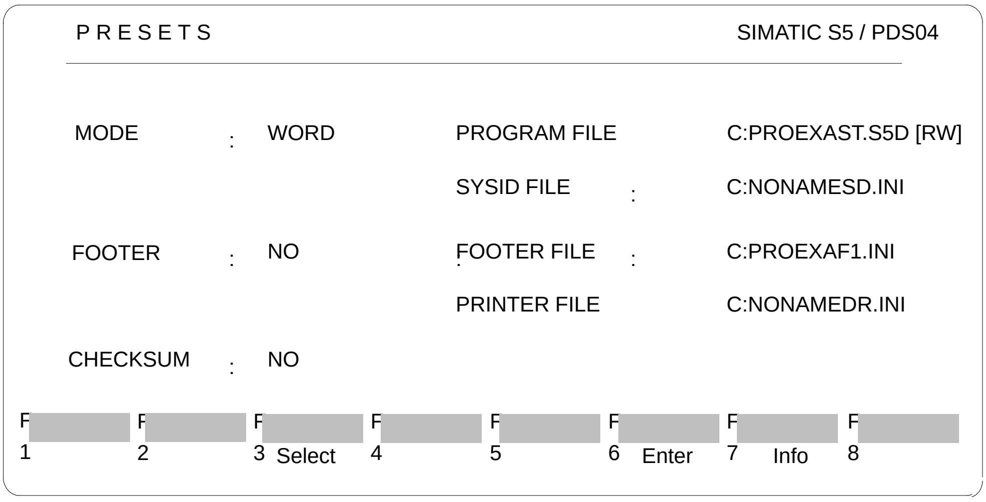
STEP 5 supports you in selecting the correct submodule parameters for different EPROM types.

The following functions are available:

- loading blocks in an EPROM/EEPROM
- reading blocks from an EPROM/EEPROM and transferring them to the active program file
- erasing EEPROM submodules
- displaying information about EPROM/EEPROMs
- transferring SYSID parameters

Note

No comment, documentation or variables blocks are transferred to the submodule.



```
PRESETS                                      SIMATIC S5 / PDS04

MODE      : WORD      PROGRAM FILE           C:PROEXAST.S5D [RW]
                      SYSID FILE    :        C:NONAMESD.INI
FOOTER    : NO        FOOTER FILE   :        C:PROEXAF1.INI
                      PRINTER FILE           C:NONAMEDR.INI
CHECKSUM  : NO
```

Operation

As soon as you have selected this function, the above box appears on the screen.

Presets

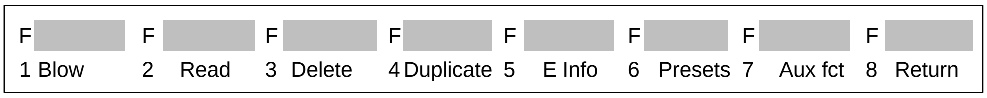
On the right-hand side of the screen, the files you have selected under → *Project* are displayed. Enter your input with ***F6*** = *Enter.* The function is then activated.

The following inputs are possible:

Input field	Explanation
Mode	
WORD	Write/read word-oriented (depending on the submodule type).
BYTE	Write/read byte-oriented (depending on the submodule type).
WORD/ FIELD	Mandatory for the CPU 946/947 (memory module 335). With the S5-155U, the first character of the useful data of a block is at the paragraph boundary (16 bytes).
Checksum NO	The checksum of the block transferred to the submodule is not created.
YES	The checksum of the block transferred to the module is created, appended to the block and transferred to the submodule.

Function Selection

Once you have entered the “presets” with ***F6***, the selection box “select function” is displayed. Your settings remain visible, but you can no longer change them. The displays simply serve as information. You can activate the individual functions using the softkey menu.



Defining the Function

How to use and define a function is described based on the “blow” function and is basically the same for the other EPROM functions (read, delete and duplicate).

Press ***F1*** = *Blow.*

The following command line is then displayed at the lower edge of the screen:

```
BLOW EPROM                    BLOCK:        PTR:
```

The following table explains the possible inputs:

Inputs		Explanation
Block		Complete this input with the ***Return*** key.
	PBn (e.g.)	Single block name.
	PB (e.g.)	All blocks of one type.
	*	A list is displayed in which you can enter a maximum of 6 blocks.
	A	All blocks in the preset program file (→ *Project*)
Ptr		You complete this input with the ***Insert*** key.
	Blank	Output only on the screen.
	*	Standard printout.
	1	Normal print.
	2	Condensed print.

Programming Number

Once you have confirmed your inputs, the following input line is displayed:

PROG NUMBER?

Here, you must enter the programming number. The programming number identifies the EPROM/EEPROM submodule you are using.

Selecting the PROG NUMBER

There are two ways of entering this number:

1. Type in the number directly.
2. Select the number using the ***HELP*** key.

A list supplied with STEP 5 contains the assignments. You can display this list with the ***HELP*** key and can page through it. You can then position the cursor on a submodule in the list and press the ***Return*** key to enter the programming number in the “PROG NUMBER” field.

The list of EPROM/EEPROM modules contains the following information:

Term	Explanation
MLFB	Order number of a module.
Prog. no.	The programmer identifies the EPROM/EEPROM submodule with this programming number. Each number belongs to a different order number.
Capacity	Memory capacity of the EPROM/EEPROM

Note

The prog.no. 500 is reserved for SIMATIC memory cards. You program and check these cards the same way as described in this section.

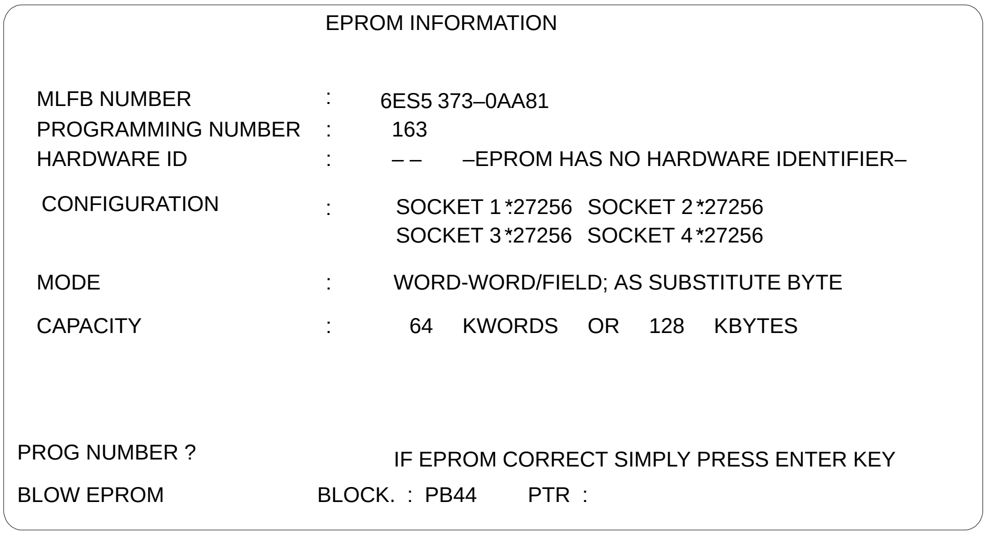
Submodule Information

Once you have typed in the programming number and pressed the ***Insert*** key, a screen containing submodule information is displayed which you also acknowledge with the ***Insert*** key.

Note

If you type in the wrong “Prog. no.”, EPROM/EEPROMs can be destroyed.

If, for example, you only type in the programming number 57 instead of 457 for submodule 6ES5 372-1AA61, the submodule will be destroyed.



```
                                   EPROM INFORMATION


  MLFB NUMBER                  :     6ES5 373–0AA81
  PROGRAMMING NUMBER           :      163
  HARDWARE ID                  :      – –    –EPROM HAS NO HARDWARE IDENTIFIER–

  CONFIGURATION                :      SOCKET 1 *27256   SOCKET 2 *27256
                                      SOCKET 3 *27256   SOCKET 4 *27256

  MODE                         :      WORD-WORD/FIELD; AS SUBSTITUTE BYTE

  CAPACITY                     :       64    KWORDS    OR    128    KBYTES



PROG NUMBER ?                         IF EPROM CORRECT SIMPLY PRESS ENTER KEY
BLOW EPROM                    BLOCK.  :  PB44      PTR  :
```

Figure 3-63 Example of the EPROM/EEPROM Information Screen

How to Activate Functions

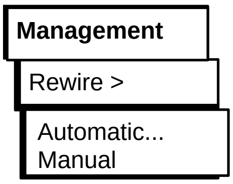
The various EPROM functions activated by softkeys (F1 - F8) are explained in the following table.

Key level 1	2	**Effect of the function keys**
		Cursor keys → *Appendix A4, key assignment*
F1		**Blow:** Transfer blocks to an EPROM/EEPROM module. Inputs are made as described on pages 3-257. The transfer is completed with the message **Main function End address nnnnnnnn** Address: The displayed addresses are physical addresses of the EPROM/EEPROM. *Cancel the transfer with ESC*: The block currently being transferred is completely transferred before the function is terminated.
F2		**Read:** Transfer blocks from an EPROM/EEPROM submodule to the active program file (→ *Project*). The transfer is completed with the message : **EPROM check Free from nnnnnnnn**
F3		**Delete:** This function erases EEPROMs and memory cards and is completed with the message: **Main function End address nnnnnnnn** EPROMs are erased with an erasing unit.
F4		**Duplicate:** only for the PC package Duplicates EPROM/EEPROM submodules using a programming unit “PROMMER”. If you do not have a PROMMER, an error message is displayed.
F5		**E info:** Displays information about the submodule inserted in the EPROM slot. Changes to the next key level.

Key level 1	 2	Effect of the function keys
	F1	**Dir:** Outputs the directory of blocks on the EPROM/EEPROM on the screen or printer. If a block or block header is found, the block list is displayed on the screen. Depending on the setting you have selected, the output is completed with the following message: For a block or a group of blocks: **Block found at** **Header end address nnnnnnnn** For all blocks: **EPROM check free from nnnnnnnn “Free from”** is the physical end address of the last block in the EPROM/EEPROM submodule.
	F2	**Compare:** Compares the S5 blocks stored in the EPROM/EEPROM with those in the active program file. The result of the comparison is displayed on the screen or printed out. During the comparison, messages appear on the screen. The following messages complete the compare function. Comparing **all** blocks: **EPROM check free from nnnnnnnn “Free from”** is the physical end address of the last block in the EPROM/EEPROM submodule. Comparing a block or a group of single blocks: **Main function End address nnnnnnnn** If there is a discrepancy between blocks, the following messages are displayed: **Address** The relative block address in the submodule. **Ref** The reference is the content of the memory location stored at the relative block address in the program file. **Act** The actual value is the content of the memory location at which the relative block address is stored in the EPROM/ EEPROM submodule.

Key level 1	 2	Effect of the function keys
	F3	**Parameters:** Output of EPROM/EEPROM parameters on the screen and comparison with the parameter values of the submodule inserted in the EPROM slot. If the information matches up, the PG displays the parameter values as shown in *Figure* 3-65.
	F5	**SYSID inp:** Transfer the data in the SYSID file to the EPROM/EEPROM submodule. If the EPROM/EEPROM submodule is not completely empty, the following message is displayed: **SYSID writing prohibited** The transfer is completed with the following message: **Main function End address nnnnnnnn**
	F6	**SYSID out:** Transfer the SYSID data contained in the EPROM/EEPROM submodule to the preset SYSID file and display on the screen. The preset SYSID file can be overwritten with this function. The transfer is completed with the following message: **Main function End address nnnnnnnn**
	F8	**Return** Return to the first key level
F6		**Presets** Return to the EPROM presets
F7		**Aux fct** Calls auxiliary functions corresponding to the user interface of STEP 5 versions prior to V6.0.
F8		**Return** Return to function selection

3.5.3 Rewiring



With the “rewiring” function, you can rename operands as follows:

- automatically, based on an assignment list or
- manually, based on a list of changes you have created.

“Rewiring” is a term going back to the days of hardwired systems and is synonymous with “rename operands”.
You want to assign different (new) addresses to one or more operands within the user program.
The rewiring function ensures that these operands are renamed in the entire user program and you only need to input the assignment once for each operand in a list. You can decide whether to address the operands in symbolic or absolute form. These operands belong to the types input I, output Q, flag F, timer T or counter C. **“S” flags are not taken into account**.

You can change the addresses but cannot change the symbol for an operand.

Blocks in which no operands have been changed are stored by STEP 5 unchanged in the “new” program file so that the structure of the user program remains the same.
To transfer the structure of the user program unchanged, the data blocks must be transferred separately to the new file.

Example

You have entered the following operands in the block: **I 1.0** (with the symbol **–MOTOR**) and **I 1.5** (with the symbol **–SWITCH**).You want to assign the new address **I 1.5** to input **I 1.0**. STEP 5 “rewires” this input (changes the address) in the required blocks or in the entire user program. If the operands in the “new” blocks have symbolic names, the symbol **–SWITCH** will be output instead of the symbol **–MOTOR**.

Note

If you replace I1.0 → I20.0, IB/EW1 does **not** become IB/IW20.

Automatic Rewiring using the Assignment List

Management
Rewire
Automatic

With this function you can rename operands automatically based on a modified or new assignment list.
You copy the assignment list (symbols file) belonging to the user program and then change the addresses of the operands in this list as required.

The PG uses this “new” assignment list as a reference list to recognize the changed operands in the entire “old” user program (or in individual blocks) **automatically** and to store the rewired operands in the second program file as a “new” user program. The “old” user program is retained. You can change any number of operands.

How to Rewire Operands

Make a copy of the assignment list and change the operands to be rewired.
After you call the automatic rewiring function, STEP 5 displays the relevant job box (→ *Graphical user interface, Job box*).

The name of the user program in which you want to rename operands is displayed in the program file field. Enter the names of the “new” files to be created as a result of the modification in the “to program file:” field and enter the file name of the copy of the assignment list in the “with new symbols file:” field.

If you only want to rename operands in certain blocks, type in the block list under “Selection” or mark all blocks of one type or all blocks.
After you click on ***OK,*** STEP 5 displays a list of the files affected by the renaming function either on the screen, printer or to a file.

Error?

If an error occurs during the renaming, the block currently being processed is not transferred to the new program file and a message displayed to this effect.

To Stop the Function

*Press **ESC** = Cancel.*
The PG does not store the block currently being processed.

Manual Rewiring with a Modification List

Management

Rewire

Manual

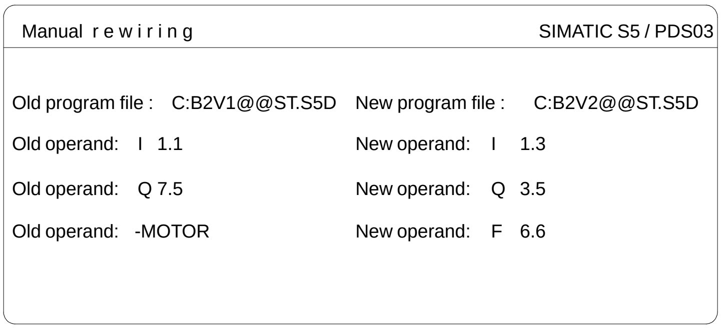
With this function you can rename operands in an operand list displayed on the screen.
Apart from the new operand addresses, you must also specify a name for the “new” user program.

After you have selected the manual rewiring function, the PG displays the relevant job box on the screen (→ *Graphical user interface, Job box*).

The name of the user program in which you want to rename operands is displayed in the “program file:” field. Enter the names of the “new” files created as a result of this modification in the “to program file:” field.

After you click on ***OK***, STEP 5 displays the empty table “Manual rewiring” in which you enter the operands in the old and new program file on the screen.
This list can contain up to 16 operands with the old and new address in absolute representation (Fig. 3-64).

Complete each entry with the ***Return*** key.



```
Manual rewiring                                              SIMATIC S5 / PDS03

Old program file :   C:B2V1@@ST.S5D    New program file :   C:B2V2@@ST.S5D
Old operand:   I  1.1                  New operand:   I   1.3
Old operand:   Q 7.5                   New operand:   Q   3.5
Old operand:  -MOTOR                   New operand:   F   6.6
```

Figure 3-64 Operand List for Manual Rewiring (Example)

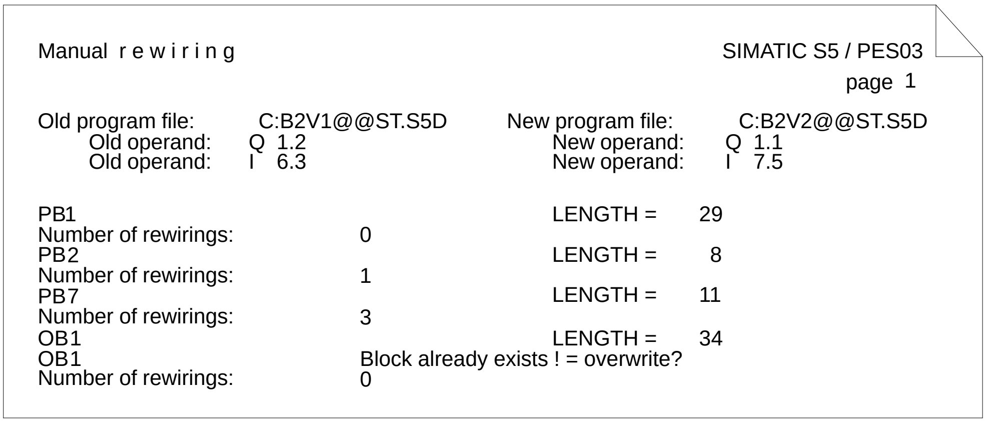
After editing the modified operand addresses, complete your input with the ***Insert*** key.

STEP 5 now renames the operands and displays the name of the block being processed in the “Manual rewiring” screen form (Fig. 3-65).

When you input the operands, STEP 5 checks each completed input field immediately for syntax errors and displays the message “syntax wrong” if an error is detected.

Printout

If you select “output on printer” in the selection box, STEP 5 prints out a list of the renamed operands after you press the ***Insert*** key. This list contains the addresses “old/new”, the number of operands renamed in the block affected in conjunction with the length information from the block header.
Error messages indicate the operand for which an error was detected. Following an error, STEP 5 aborts the rewiring function.



```
Manual rewiring                                                    SIMATIC S5 / PES03
                                                                             page  1

Old program file:      C:B2V1@@ST.S5D        New program file:      C:B2V2@@ST.S5D
     Old operand:     Q  1.2                      New operand:     Q  1.1
     Old operand:     I   6.3                     New operand:     I   7.5

PB1                                               LENGTH =     29
Number of rewirings:            0
PB2                                               LENGTH =      8
Number of rewirings:            1
PB7                                               LENGTH =     11
Number of rewirings:            3
OB1                                               LENGTH =     34
OB1                             Block already exists ! = overwrite?
Number of rewirings:            0
```

Figure 3-65 Printout Following Manual Rewiring (Example)

To Stop the Function

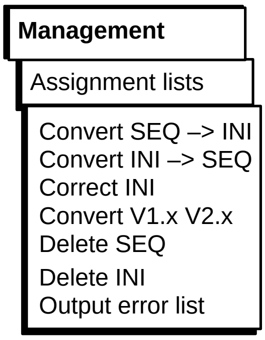
Press ***ESC*** = *Cancel*

The PG does not store the block currently being processed.

Error?

If an error occurs during rewiring, the block in which the error occurs is not transferred to the “new” program file and a message is displayed to this effect.

3.5.4 Assignment Lists



With this function you process the assignment lists required to address operands symbolically in your user programs.

The following functions are available:

- Translation of a sequential source file into a symbols file (*Z0.SEQ → *Z0.INI).
- Translation of a symbols file into a sequential source file sorted according to absolute operands or symbolic operands (*Z0.INI → *Z0.SEQ) with or without sorting the operands.
- Fast correction of the assignment list directly in the translated symbols file (*Z0.INI).
- Translation of an old symbols file into a sequential source file (Convert stage V1.x V2.x).
- Deleting a sequential source file with the corresponding error file.
- Deleting a symbols file.
- Outputting the list of translation errors (error file).

In the PLC, operands are only processed with absolute addresses. As a result, the assignment of a “symbolic address” to an “absolute address” (e.g. button 1 → I 1.1) always requires an assignment list with a symbols file (*Z0.INI) derived from it.

Editing an Assignment List

How to edit an assignment list is described in Section 3.3.12. The source file (*Z0.SEQ) generated following editing, is converted into three symbols files (*Z0.INI, *Z1.INI, *Z2.INI) following translation.

Generating Symbols Files

The symbols files are generated automatically by STEP 5 after you call the function “Convert SEQ → INI” or when you edit the assignment list.

Processing in the PLC

To translate the user program so that it is suitable for the PLC when it is loaded, only the symbols files are required.

Converting SEQ → INI

Management

Assignment lists

Convert SEQ –> INI

With this function, you translate the sequential source file into the corresponding symbols file.

After selecting “Convert SEQ → INI”, STEP 5 displays a job box in which you type in the name of the source file to be translated, if it is not already displayed.
After you click on ***OK***, the file is translated.

If you have included absolute operands without corresponding symbolic operands in the sequential source file, the message “Accept absolute operand as symbol?” is displayed.
Acknowledge this message either with ***yes*** or ***no***.

If the conversion is error-free, the message “n lines processed, no error found” is displayed which you confirm with ***OK***.

If errors occur during the conversion, the message “n lines processed, x errors found” is displayed. Once again acknowledge this message with ***OK***.

Converting INI → SEQ

Management

Assignment lists

Convert INI –> SEQ

With this function, the symbols file is converted to the corresponding sequential source file, sorted according to absolute or symbolic operands as you require.

After you select the function “Convert INI → SEQ”, STEP 5 displays a job box in which you type in the name of the symbols file to be translated and specify how the source file is to be sorted.
After clicking on ***OK***, the file is translated.

The translation is completed with the message “n lines processed, no errors found” which must be acknowledged with ***OK***.

Correcting Assignments in the Symbols File

Management

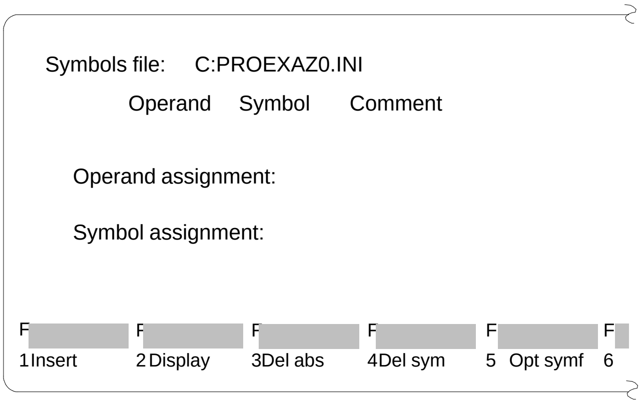
Assignment lists

Correct INI

With this function, you can correct individual assignments in long assignment lists (avoiding long translation times required for all the assignments).

After selecting the function “Correct INI”, STEP 5 displays a job box in which you can type in the name of the symbols file to be corrected if the name you require is not already displayed.

After clicking on ***OK***, the following box is displayed:



Symbols file: C:PROEXAZ0.INI

Operand Symbol Comment

Operand assignment:

Symbol assignment:

F 1	F 2	F 3	F 4	F 5	F 6
Insert	Display	Del abs	Del sym	Opt symf	

Inputting the Assignment Line

Below the three terms “Operand - Symbol - Comment” there is an input line. Here, you type in a new assignment in the symbols file.

The cursor is positioned at the beginning of the input line. The input line is edited in the “overwrite” mode.

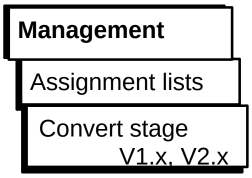
- The ***DEL*** key deletes the character marked by the cursor.
- The ***horizontal expand*** key inserts a blank at the cursor position.
- With the ***roll screen*** (up and down) keys you can alternate between input and display lines.
- The ***Return*** key and the ***TAB*** key move the cursor one input field to the right.

When editing the assignments in the symbols file, STEP 5 makes the following functions available with the function keys.

Function	Explanation
F1 *= Insert*	The assignment in the input line is entered providing the operand address is not assigned. Otherwise, the error message: “Key already exists” is displayed.
F2 *= Display*	The assignment to the absolute or symbolic parameter is displayed if this exists in the symbol file. The display remains on the screen until you press ***F2*** again.
F3 *= Del abs*	The assignment belonging to the absolute parameter (operand) in the input line is deleted from the symbols file. If the assignment is not defined, an error message is displayed.
F4 *= Del sym*	The assignment belonging to the symbolic parameter in the input line is deleted from the symbols file. If the assignment is not defined, an error message is displayed.
F5 *= Opt symf*	The assignment list is optimized.
F8 *= Return*	After modifications in the symbols file, STEP 5 prompts you to confirm that the source file (Z0.SEQ) should be generated. If you want to generate the source file, press the ***Insert*** key, otherwise terminate with ***NO***.

1. If you want to insert a new operand in the symbols file,
   *type in a free absolute and symbolic address and the operand comment and press **F1** = Insert.*
2. If you want to rename the absolute address of an existing operand,
   *type in the relevant operand and delete its absolute address with **F3** = Del abs. Now overwrite the operand with its new address and press **F1**.*
3. If you want to change the symbolic address of an existing operand,
   *proceed as described under 2), but delete with **F4** = Del sym.*

Converting Stage V1.x V2.x



Management

Assignment lists

Convert stage V1.x, V2.x

The byte address of an absolute parameter in the “old” assignment list of the S5-DOS software V1.x and V2.x under PCP/M is three bytes long. In STEP 5 under S5-DOS/ST/MT, the byte address is four bytes long owing to the introduction of new flags (S). For this reason, the “old” symbols file must be translated to a “new” source file before you can work with it.

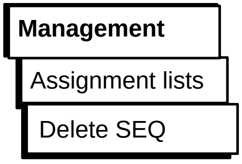
Assignment lists created with V3.x do not need to be converted.

Type in the name of the sequential source file in the displayed job box. When you click on ***OK***, the file is translated.

During the translation, a message is displayed on the screen.

If you have specified absolute operands without corresponding symbolic operands in the sequential source file, a message is displayed. Acknowledge the message to suit your requirements.

Deleting SEQ



Management

Assignment lists

Delete SEQ

With this function you can delete a sequential source file.
At the same time, the error list assigned to the file is also deleted.

After you trigger the function “Delete SEQ”, STEP 5 displays a job box in which you type in the name of the source file to be deleted if it is not already displayed.

After clicking on ***OK***, the *SEQ files are deleted.

On completion of the function, the deleted files are listed on the screen.

Deleting INI

Management
Assignment lists
Delete INI

With this function, you delete the symbols files (*Z0.INI, *Z1.INI and *Z2.INI).

After selecting the function “Delete INI”, STEP 5 displays a job box in which you type in the name of the symbols file to be deleted if this is not already displayed.

After clicking on ***OK***, the symbols files are deleted. On completion of the function the deleted files are listed on the screen.

Outputting the Error List

Management
Assignment lists
Output error list

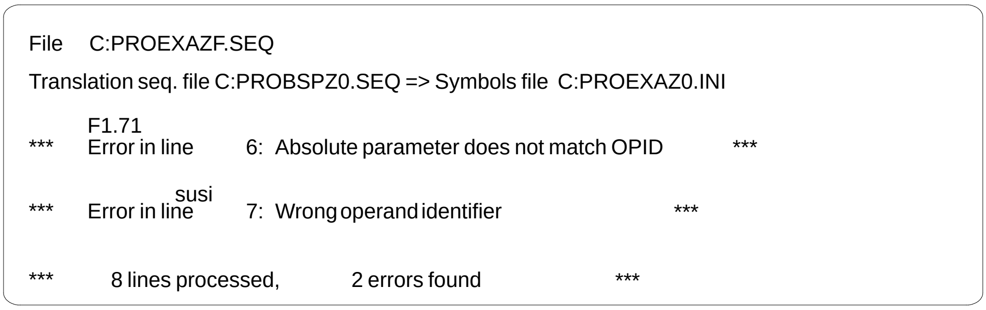
STEP 5 collects the error messages occurring during one of the following translations.

- Translation of the sequential file *Z0.SEQ into the symbols files (*Z0.INI, *Z1.INI, *Z2.INI)
- Retranslation of the symbols files into the sequential source file (INI → SEQ).

After calling the function “Output error list” a job box is displayed in which you type in the name of the error file (*ZF.S5D) to be output, i.e. the name of the program file but with the extension *ZF.S5D.

Example

After clicking on ***OK***, STEP 5 displays the error file.



```
File    C:PROEXAZF.SEQ
Translation seq. file C:PROBSPZ0.SEQ => Symbols file  C:PROEXAZ0.INI

        F1.71
***     Error in line      6:  Absolute parameter does not match OPID        ***

                    susi
***     Error in line      7:  Wrong operand identifier              ***


***        8 lines processed,         2 errors found              ***
```

Figure 3-66 Error List after Editing the Source File (Example)

An error message indicates the incorrectly assigned operand, the location of the error and the error type.

Each time you translate the same sequential source file, STEP 5 automatically overwrites the previously stored error list. The file is also generated if no error occurs.

3.5.5 Selecting a Drive

Management
Select drive

Using this function, you select the drives in which STEP 5 searches for files with S5 system blocks. The selected drive (marked with an asterisk) from which STEP 5 is currently active is displayed but cannot be changed.

The “drives” selection box is displayed. The cursor flashes on the specified drive. You can move the cursor with ***cursor left*** or ***cursor right***. After you have selected a drive with ***F3***, you must save the setting with ***F6***.

3.5.6 Bus paths

Management
Bus paths

Online connections between programmers and the modules of the PLC are not only established by direct connecting cables (point-to-point connection) but also via the bus systems SINEC H1, SINEC L1 or SINEC L2 and the PLC bus (with the S5-155U).

You can create, store and activate these connections with the “bus paths” function. When editing, STEP 5 supports you graphically.

– With this function, you edit and activate **paths**.
  Paths are permanent connections from a PLC to a station. Via this path, you can perform all the programming functions according to the protocol just as with a direct point-to-point connection.

  A path consists of the following:

  - start node. (e.g. PG/AS511, PG/CP-H1. PG/CP-L2),
  - bus (1 or more)
  - nodes (e.g. CP),
  - end nodes (e.g. CPU)

– You edit and store **station addresses** in the offline mode.

– An edited path is stored under a **path name** (→ *Project, settings*) and this can be activated at any time provided it exists physically.

– You can store several paths with their path names in a selectable **path file** (→ *Project, settings*) and activate a path using its name.

– The **establishment** (activation) of a path is supported. This is, however, only possible in the online mode.

– The **termination** (deactivation) of a path is supported by this function.

You can assign 4 files to each path:

- Program files....ST.S5D
- Symbols files....ZO.INI
- Printer files....DR.INI
- Footer files....F1.INI or F2.INI

These file names are saved along with the path in the path file. The assignment does not affect existing files. You can also assign files that do not yet exist and that you will create later. By assigning files to a path, you do not change the project settings. To set these files in the current project, you must select the path in the project settings (set the path option to *always* or *confirmation*).

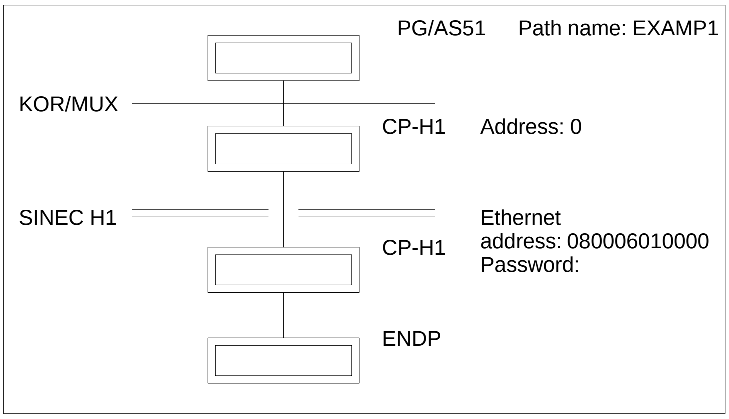


Figure 3-67 Example of an Edited Path

Settings

The AS511 interface must be set.
For more information about settings refer to → *Project*.

Operation

After selecting the “Bus Paths” function, the selection box “Select function/presets” is displayed. If you have not yet made these settings (→ *Project*), you can set the following:

- path file
- path name

Editing (Files for the Path)

After selecting the Files function, the four file entries for the current path are displayed. You can edit these and save them again. You can enter any file names you wish. With a new path or after deleting the file entries, only the file name extensions are displayed.

Key level 1	2	**Explanation**
F3		**Select** The “file selection” box is displayed. This lists the existing files of the various types (depending on the cursor position). You can select one of these and activate it with ***OK*** **Cursor on the “Path name” input field** A box is displayed containing all the paths in the selected path file. You can activate the path marked by the cursor with ***OK***.
Shift F3		**Delete** The four file entries for the path are deleted. No existing files are modified, but rather the assignment between the path and the files is cancelled.
F4		F1 ⇔ F2 If the cursor is located in the input line for the footer file, you can change over between footer files ...F1.INI (80 characters wide) and ...F2.INI (132 characters wide). If you call the function *F3 = Select*, footer files corresponding to the current setting are listed.
F7		**Enter** You buffer the file entries made up to now and return to the menu. The file entries are only saved in the path file when you save the path.
F8		**Cancel** Cancels the editing and you return to the menu. All changes you have made are discarded.

Setting Bus Paths

The inputs you can make in the “Select function/presets” selection box are described in the following table (example, see page 3-281).

Key level		Explanation
1	2	
F1		Edit: The path editor is started. You can now edit the bus path in the working field displayed. The softkeys are assigned a new function. Owing to the wide range of functions of the bus editor, you will find detailed information from page 3- 272 onwards.
F2		Print You branch to the “documentation” function level.
	F3	Dir: Prints the (path) directory of the specified “path file”.
	F4	All paths: Prints all the path names in the specified “path file”.
	F5	Cur path Prints the currently set path in the “path file”.
	F8	Return: Returns to the last menu (select function).

Key level 1	2	**Explanation**
F3		Select: **Cursor positioned on the “path file” input field.** The file selection box is displayed. This lists all the path files. You can select a path from this list and enter it with ***OK***. **Cursor on the “path file” input field.** A box is displayed listing all the paths in the set path file. You can enter the path marked by the cursor with ***OK***.
F4		Set up: With this function, the set path is displayed. You can correct through to the end point step by step using the function F3 = Next n(ode) or in one step (***F5*** “all n(odes”). Selected nodes are marked by “*”. With the CPs (H1, L2 and L1) you can read out the system identification with ***F1*** (n SYSID). This data cannot be modified.
F5		Terminate: The connection set up with ***F4*** is terminated in the order determined by the path.
F6		Delete: The path name is deleted in the selected path file.
F8		Return: Return to the last menu. You exit the bus path function.

Editing Bus Paths

F1

This starts the bus path editing function. Here, you have two possibilities:

1. **The path name exists.**
   The path is displayed in the path field. You can delete the nodes one by one using ***F7***, beginning with the last node. Use the softkeys to insert new nodes.
2. **You are creating a new path.**
   By specifying selectable nodes one after the other you can create a path to suit your system. If you select an unsuitable path configuration, the message **“not pref. path”** is displayed.

Note

The path is set up even if the message “not pref. path” appears. Siemens, however, cannot guarantee that such path will function.

Selecting Nodes

If you press one of the function keys displayed in the menu, a corresponding node is displayed graphically. You then change to a new function key level. Here, you can select a further node or bus. Within these function key levels, only nodes or buses suitable for the configuration you have selected are available.

Node addresses:
Each node has an address assigned either by jumper or switch settings or assigned using the software. The bus editor recognizes two node addresses:

- **Address** (KOR/MUX, SINEC L1 and SINEC L2). When you edit, you must type in the address in decimal in the “address field”.
  - KOR/MUX address from 1 to 30.
  - SINEC L1 address from 1 to 30
  - SINEC L2 address from 1 to 31
- **Ethernet address**. This only occurs with SINEC H1 bus system, it must be entered in hexadecimal.

Start Node

You can select the following start nodes at the highest key level of the editor:

F2	PG/AS511	
F3	PG/CP-H1.	In this editing mode, the PLC bus and end node PG/CP-H1 can be selected.
F4	PG/CP-L2	

During editing, these start nodes are not dependent on the set interface. The functions of the softkeys from now on depend in part on the start node you have selected.

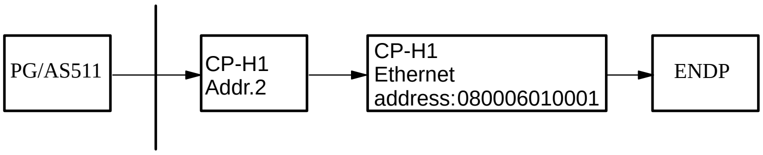
Function Keys

In the editing mode (***F1***) the function keys are assigned as follows for all function levels:

Function	Explanation
F1 = *ENDP*	Add the end node (end point).
F2 = *KOR-MUX*	Add a bus of the type AS511.
F3 = *CP-H1*	Add a node of the type CP-H1.
F3 = *PLC-BUS*	Add a bus of the type PLC bus (backplane bus). This is only possible with the S5-155U.
F3 = *PG/CP-L2*	End node of the type PG/CP-L2.
F4 = *CP-L2*	Add a node of the type CP-L2.
F4 = *PG/CP-H1*	Add an end node of the type PG/CP-H1.
F5 = *CP-L1*	Add a node of the type CP-L1.
F6 = *Enter*	The edited path is stored. The selection box displays “save path as”. 1. Here, specify a path file and a path name. The path is stored in the path file (press ***F6***). Here, you select a path file. 2. With ***F3*** and the cursor on the “path file” input field, the selection box “file selection box” is displayed with all its path files. 3. With ***F3*** and the cursor on the “path name” input field, a selection box is displayed containing all the paths in the selected path file. Here, you can select a path name.
F7 = *Del elem*	Deletes the last node and/or bus from the path.
F8 = *Spec fct*	This function is for paths created with STEP5 stage 5. Entries in the displayed “function info” have no effect.
SHIFT F8 = *Help*	Displays information about the functions of the softkeys at the current level.

Editing Example

Editing a path.
You want to create the following path:



KOR/MUX with address 7

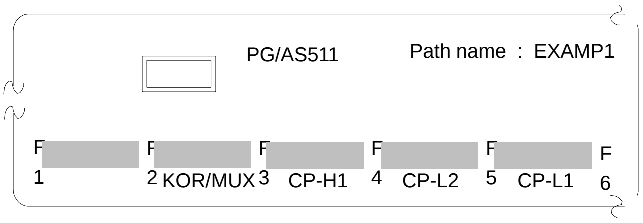
Ready to Start?

The AS511 interface is set.
You have selected the function “Management, Bus paths”.

Operation

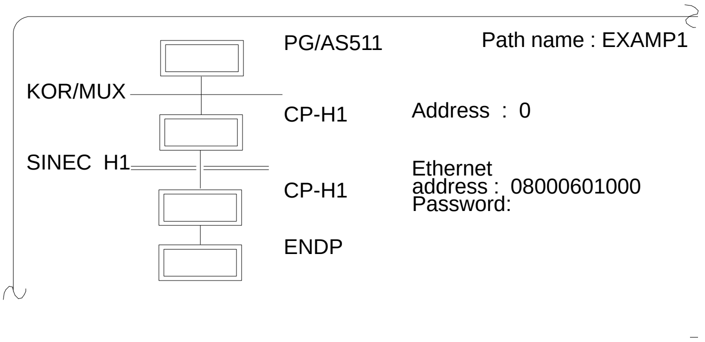
The “select function/presets” box is displayed.

1. Specify the path file.
2. Type in a new path name.
3. Press ***F1*** = *Edit.*
   An empty working field is displayed along with the following softkeys:
   F2 = PG/AS511
   F3 = PG/CP-H1
   F4 = PG/CP-L2
4. Press ***F2*** = *PG/AS511*.
   The start node is displayed and the softkeys are assigned new functions.



5. Press ***F2*** = *KOR/MUX*
   The KOR/MUX bus is added and the softkeys are assigned new functions.

6. Press ***F3*** = *CP-H1*
   The CP-H1 node with the SINEC H1 bus is added and the softkeys are assigned new functions.
7. Press ***F3*** = *CP-H1*
   The CP-H1 node is added and the softkeys are assigned new functions.
8. You can now type in the MUX address, the Ethernet address and if required the password. Move the cursor to these fields using the ***cursor down*** key.
9. Press ***F1*** = *ENDP*
   The end point, i.e. the destination of the bus connection, is added. New function key functions are displayed. The screen should appear as follows when the path is complete.



Path name : EXAMP1

Address : 0

Ethernet address : 08000601000
Password:

The path must now be stored.

10. ***Press F7*** = *Enter.*
    If the file name is new, a box appears in which you once again have the opportunity of changing the name.
11. *Press **F7** = Enter.*
    The path is stored in the path file and you can activate it at any time.

3.6 Documentation

The documentation menu provides a range of functions with which you can output program sections such as blocks, files and lists on a printer (A3, A4) or to a file, e.g.

- program blocks, data blocks, lists, structures
- text files (ASCII files)

In addition to this, it is also possible to evaluate certain data according to different criteria, e.g.

- output the cross reference list according to selected operands
- output the assignment list sorted according to symbolic operands

If the screen display covers more than one page, the prompt “Continue? yes/no” is superimposed on the screen. You can clear this box from the screen with the space bar.

You can add a footer to the bottom of each printout. Outputs are either printed out or are directed to a file.

You can make **hardcopies** of the screen at any time. Once again, these can be printed out or stored in an ASCII file.

The following functions can be selected:

- Preparing for printing (settings)
  You set the printer parameters and footer
- Standard output
  The program sections are output in the form in which you edited them and with a footer if you have selected this function. The data can be output either from the program file or from the PLC.
- Enhanced output
  The program sections are printed out with additional graphical elements (lines, boxes etc.) and a footer. This data can only be output from the program file and not directly **from the PLC**.

- Doc command
  All the functions of the enhanced output can be executed by doc commands which you edit and store in files. Using these commands, you can run frequently recurring outputs without laborious input routines. Some doc commands can be used to call further doc command files achieving a sequential structure. This can be represented graphically with the “Edit structure” function.

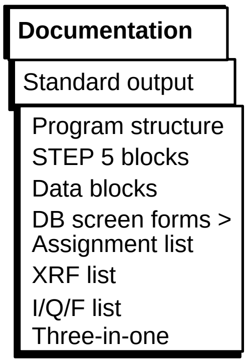
Getting to Know the Functions

This chapter is structured so that the possible inputs and selections are explained for the various functions, whereas the key functions themselves are described extra in a separate section (→ *Graphical user interface, Job box*).

To get to know the documentation functions quickly, we recommend the following procedure:

1. Familiarize yourself with the key functions described in → *Graphical user interface*.
2. The inputs required for each individual function are explained in this chapter. If you are familiar with the key functions, you will then find it easy to activate these functions.

3.6.1 Standard Output



With this function, you can output program sections in their basic form (as you edited them) either on a printer (A3, A4) to files or on the screen. You can decide whether to output from the program file or from the PLC. You can output the following program sections:

→ ***program structure***
→ ***STEP5 blocks***
→ ***data blocks***
→ ***DB screen forms***

You can also output the following lists:

→ ***assignment list***
→ ***XRF list***
→ ***I/Q/F list***
→ ***three-in-one (all three lists at once)***

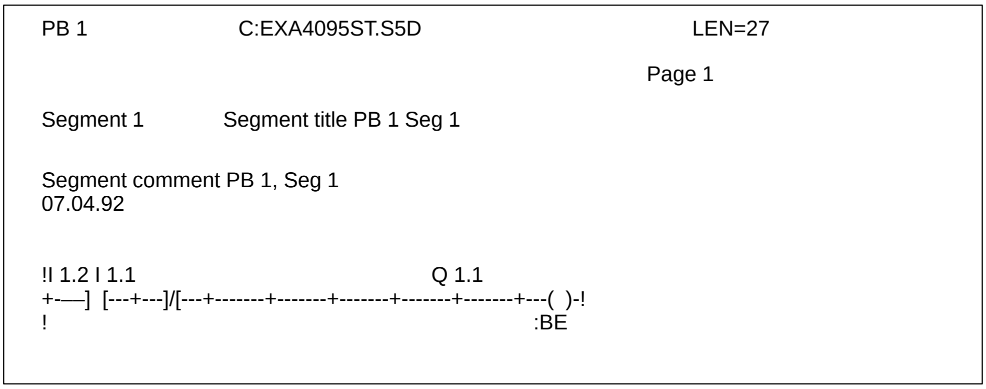
Note

For standard output, no cross reference list (file *XR.INI) is necessary.

You can output this information both from a program file as well as from a PLC. In this case, the PLC type and CPU ID must also be specified in the selection boxes.

Example of a Printout

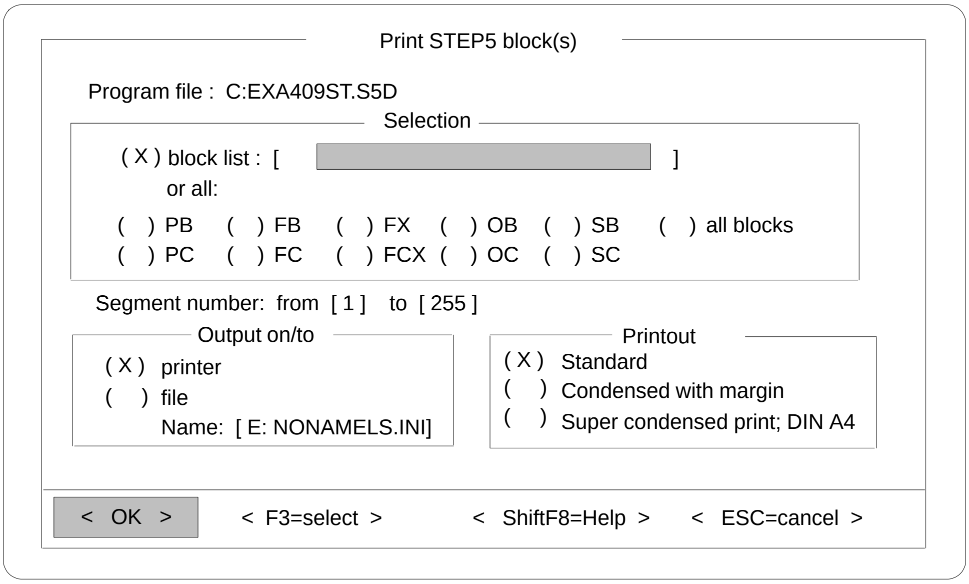
The following example in the LAD method of representation (PB1, segment 1) contains a STEP 5 block in its basic form, i.e. the blocks are printed out as you edited them. If you select enhanced output, further graphical information is added to the printout. The footer is not illustrated.



```
PB 1                 C:EXA4095ST.S5D                              LEN=27

                                                               Page 1

Segment 1          Segment title PB 1 Seg 1


Segment comment PB 1, Seg 1
07.04.92


!I 1.2 I 1.1                                  Q 1.1
+-—]  [---+---]/[---+-------+-------+-------+-------+-------+---(  )-!
!                                                    :BE
```

Operation

Once you have selected the functions for outputting program sections and lists, a job box is displayed with a structure similar to that for standard functions (→ *Graphical user interface, Job box*).



Print STEP5 block(s)

Program file : C:EXA409ST.S5D

Selection

(X) block list : [ ]

or all:

( ) PB ( ) FB ( ) FX ( ) OB ( ) SB ( ) all blocks

( ) PC ( ) FC ( ) FCX ( ) OC ( ) SC

Segment number: from [1] to [255]

Output on/to

(X) printer

( ) file

Name: [E: NONAMELS.INI]

Printout

(X) Standard

( ) Condensed with margin

( ) Super condensed print; DIN A4

Figure 3-68 Example of a Job Box

Program Structure

Documentation

Standard output

Program structure

With this function, you can output the call structure (program overview) of the individual blocks in a user program. You can output the program overview from the program file or from the PLC. The output is in three parts:

1. List of all blocks (including symbolic names if they exist) including the length, number of words of the individual blocks.
2. List of all block types in the program file, with the length of each block type.

3. Program overview in which the nested calls (nesting depth maximum 8 block calls) of the individual blocks starting with the block type OB is specified. With each block, a further ID is output as follows:

–	Block is called unconditionally
=	Block is called conditionally
#	Block call follows a BO DW or DO FW operation (indirect addressing)
?	Block call as formal operand An actual operand can be output as a constant or as MC 5 machine code.
???????	The called block does not exist in the program file
!F113!	There are further block calls that cannot be represented (nesting depth too great)
!F114!	Recursive block call, e.g. calling an OB in a PB

You can display the structure on the screen and output it to a printer or to a file.

Settings

The following must be set:

- program file
- symbols file (only when “symbols:yes” is set)
- footer file (only if a footer is required)
- mode (online when you want to output from the PLC)
- printer file (the PT88 is the default printer)

Operation

Information about making settings can be found in → *Project*. The job box “Output program overview” is displayed. Here, you can make your selections (→ *Graphical user interface*).

Example

Standard output of a program structure with data blocks.

```
  Program overview with DB                                              Page  1
 PB              1    :                               Length  :          9
 PB              2    :                               Length  :          21
 PB              3    :                               Length  :          9
 PB             12    :                               Length  :          25
 FB             10    :                               Length  :          50
 OB              1    :                               Length  :          13
 DB             10    :                               Length  :          28
 Length  :  PB               64
 Length  :  SB                0
 Length  :  FB               50
 Length  :  FX                0
 Length  :  OB               13
 Length  :  DB               28
 Length  :  DX                0
 Length  :                  155
```

```
 Program overview with DB                                              Page   2
+–OB   1– +=PB    1– +DB    10–
          I          I
          I          +=PB    3–+FB 10
          I          I
          !          !
          .          .
```

Figure 3-69 Program Overview with DB

STEP 5 Blocks

Documentation

Standard output

STEP 5 blocks

With this function, you can output the blocks of a program file or from the PLC memory in the LAD, CSF or STL method of representation. You can also output the blocks to a file or printer.

Settings

The following must also be set:

– program file
– footer file (only if a footer is required)
– symbols file (only when “symbols: yes” is set)
– mode (online when you want to output from the PLC)
– printer file (the defaults apply to the PT88)

For more information about settings, refer to → *Project*.

Operation

The job box “Print STEP 5 block(s)” is displayed. Here, you can make your selections (→ *Graphical user interface*). In the following table only the inputs for this function are explained.

Input field	Explanations
Segment number from to	Output segment numbers from n to n from a program block.
STL address rep.	Only when STL is selected: select the type of address information.
None	No addresses in the printout.
Byte oriented	Statement address output in byte
Word oriented	Statement address output in words.

Data Blocks

Documentation

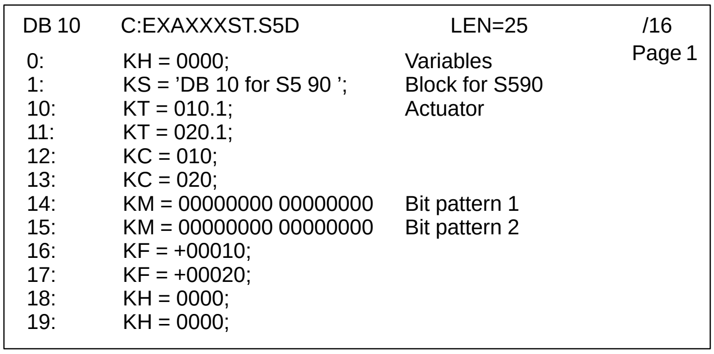
Standard output

Data blocks

With this function, you can either output individual or all the data blocks of a program.

Example of an output

”With comments” was selected in the “settings”.



```
DB 10     C:EXAXXXST.S5D                    LEN=25          /16
                                                           Page 1
 0:        KH = 0000;                Variables
 1:        KS = 'DB 10 for S5 90 ';  Block for S590
 10:       KT = 010.1;               Actuator
 11:       KT = 020.1;
 12:       KC = 010;
 13:       KC = 020;
 14:       KM = 00000000 00000000    Bit pattern 1
 15:       KM = 00000000 00000000    Bit pattern 2
 16:       KF = +00010;
 17:       KF = +00020;
 18:       KH = 0000;
 19:       KH = 0000;
```

Figure 3-70 Example of Data Block Output

Settings

The following must also be set:

- program file
- comments
- footer file (only if a footer is required)
- mode (online when you want to output from the PLC)
- printer file (the defaults apply to the PT88)

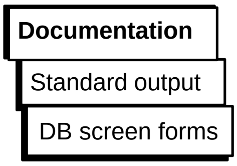
For more information about settings, refer to → *Project*.

Operation

The job box “Print STEP 5 data block(s)” is displayed. Here, you can make your selections (→ *Graphical user interface, Job box*).

Here, only the inputs specific to this function are explained.

DB Screen Forms



With this function, you can output data blocks containing screen forms.

The following must be set:

- program file
- footer file (only if a footer is required)
- mode (online when you want to output from the PLC)
- printer file (the defaults apply to the PT88)

For more information about settings, refer to → *Project*.

Operation

The job box “DB SCREEN FORMS: print blocks” is displayed. Here, you can make your selections (→ *Graphical user interface, Job box*).

Assignment List

Documentation
Standard output
Assignment list

With this function, you output an assignment list.

Example of an Output

```
File C:EXA409Z0.SEQ

  Operand            Symbol            Comment
   I 1.1              INP 1             Input 1.1
   I 1.2              INP 2             Input 1.2
   I 1.3              INP 3             Input 1.3
   I 2.1              S  2–1            Input 2.1
   .                  .                 .
   .                  .                 .
   .                  .                 .
```

Figure 3-71 Example: Output of an Assignment List

Settings

The following must be set:

- – footer file (only if a footer is required)
- – printer file (the defaults apply to the PT88)

For more information about settings, refer to → *Project*.

Operation

The job box “Print SYMBOLS:SEQ file” is displayed. Here, you can make your selections (→ *Graphical user interface, Job box*).

XRF List

Documentation
Standard output
XRF list

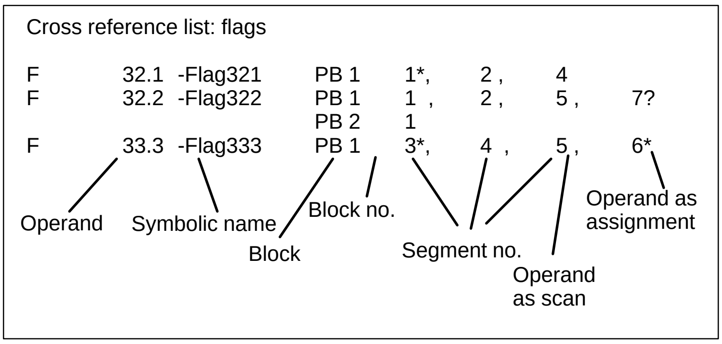
With this function, you output a cross reference list from an existing program file.

The following information is provided:

- – cross references to operand areas I, Q, F, T, C. **S flags are not output in the cross reference list**.
  cross references to data
  cross references to I/Os
  cross references to block calls
- – cross references to individual symbolic or absolute operands (e.g. –MOTOR, I1.0)
  You cannot specify a single operand only for a single block.

The cross reference list contains

- the blocks processed
- the cross references sorted according to operand, symbolic name, the blocks and segments and an identifier (see table).



```
Cross reference list: flags

F          32.1  -Flag321     PB 1    1*,     2 ,     4
F          32.2  -Flag322     PB 1    1  ,    2 ,     5 ,     7?
                              PB 2    1
F          33.3  -Flag333     PB 1    3*,     4  ,    5 ,     6*
```

The significance of the “identifier” beside the segment is as follows:

Identifier	Explanation
”Blank”	The operand occurs as a scan (e.g.: -A I 1.0)
*	The operand occurs as an assignment (e.g.: Q 1.1).
?	The operand occurs as a parameter for an FB call. An actual operand can be output as a constant or as MC 5 code.
#	The operand follows DO FW or DO DW operations (indirect addressing).
S	The operand is addressed in a standard function block.
!	The operand is addressed in a standard function and in a user block.
^	Operand references continued.

Settings

The following must be set:

– program file

– symbols file (only if symbols are selected in the presets)

– footer file (only if a footer is required)

– mode (online when you want to output from the PLC)

– printer file (the defaults apply to the PT88)

Operation

After you call the “XRF list” function, the “Output cross reference list” job box is displayed. Here, you can make your selections
(→ *Job box*). In the following table only the inputs specific to this function are explained.

Input field	Explanations
Cross reference list	
all elements	All elements (operands) are listed in the order I, Q, F, S, T, C, B, P, D on one page.
Flags, data block, inputs, timers, I/Os, outputs, counters, block calls	A cross reference list is only output for these elements.
Single operand	Indicates the occurrence of an operand in all blocks. If you only specify a single block, an error message is displayed. ***F3*** = *Select* is not possible in this situation.

I/Q/F list

Documentation

Standard output

I/Q/F list

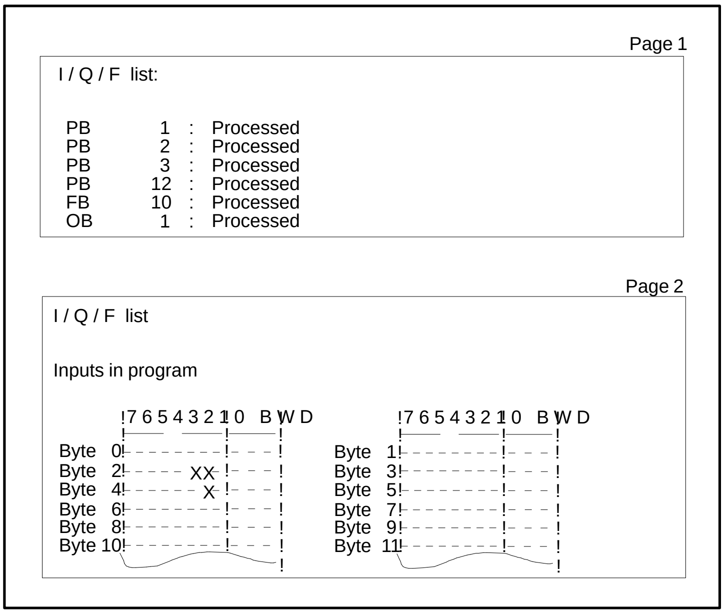
With this function, you output an I/Q/F list on the screen, on a printer or to a file. The I/Q/F list takes the form of a table and provides you with an overview of which bit is occupied in the I, Q, F, operand areas. One line is reserved for every two bytes of an operand area, in which the 8 possible bits are marked.

- a byte (**B**)
- a word (**W**)
- a double word (**D**)

(see Fig. 3-72)

Meaning of the identifiers in an I/Q/F list:

Identifier	Explanation
”Blank”	The operand is addressed as a byte, word or double word operation and not as a bit operation.
–	The operand is not addressed.
X	A bit operation is performed on the operand.
#	The operand follows DO FW or DO DW operations.
S	The operand is addressed in a standard function block.
?	The operand occurs as a parameter of an FB call.
!	The operand is addressed in a standard FB and in a user FB.



```
                                                        Page 1
 I / Q / F  list:


  PB        1  :  Processed
  PB        2  :  Processed
  PB        3  :  Processed
  PB       12  :  Processed
  FB       10  :  Processed
  OB        1  :  Processed

                                                        Page 2
 I / Q / F  list


 Inputs in program

           !7 6 5 4 3 2 1!0   B W D           !7 6 5 4 3 2 1!0   B W D
           !—————   —————!—————!               !—————   —————!—————!
 Byte  0!- - - - - - - -!- - - !       Byte  1!- - - - - - - -!- - - !
 Byte  2!- - - - - X X -!- - - !       Byte  3!- - - - - - - -!- - - !
 Byte  4!- - - - - - X -!- - - !       Byte  5!- - - - - - - -!- - - !
 Byte  6!- - - - - - - -!- - - !       Byte  7!- - - - - - - -!- - - !
 Byte  8!- - - - - - - -!- - - !       Byte  9!- - - - - - - -!- - - !
 Byte 10!- - - - - - - -!- - - !       Byte 11!- - - - - - - -!- - - !
                             !                                     !
```

Figure 3-72 Example of a Standard I/Q/F List

Settings

The following must be set:

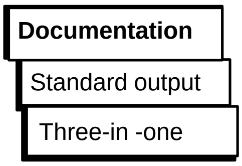
- program file
- footer file (only if a footer is required)
- mode (online when you want to output from the PLC)
- printer file (the defaults apply to the PT88)

For more information about settings, refer to → *Project*.

Operation

The job box “Display I/Q/F list” is displayed. Here, you can make your selections (→ *Graphical user interface*).

Three-in-one


Documentation
Standard output
Three-in -one

With this function, you trigger a multifunction job in which

- program overview
- I/Q/F list
- XRF list

are output one after the other without interruption, either on the screen, to the printer or to a file.

Settings

The following must be set:

- program file
- symbols (if you require symbolic representation)
- footer file (only if you require a footer)
- mode (online when you want to output from the PLC)
- printer file (the defaults apply to the PT88)

For more information about settings, refer to → *Project*.

Operation

The job box “Execute three-in-one” is displayed. Here, you can make your selections (→ *Graphical user interface, Job box*).

Note

For standard output, no cross reference list (file *XR.INI) is necessary.

3.6.2 Enhanced Output

Documentation
Enhanced output
Program sections
Reference data
Text files
Doc commands

The “enhanced output” function, previously also known as KOMDOK allows you to document STEP 5 and GRAPH 5 programs in detail and for the most part automatically (using doc commands). In contrast to the standard output, program data can be sorted and evaluated and also prepared **in a graphical form**. Output is also possible with continuous lines (see Figure 3-73 and Figure 3-74). You can print on either A3 or A4 paper. The printout on A4 paper is a compressed form of the A3 printout. The objects to be documented must be located on diskette or hard disk. If you only have programs in the PLC memory, you must first transfer them from the PLC to diskette or hard disk.

The main feature of the enhanced output is that you can use → *Doc commands* to control the printout with a minimum of keystrokes. There are doc commands for all the functions of the enhanced output. You can store doc commands in a selectable file.

You can output or generate the following:

- ***STEP 5 blocks*** and GRAPH 5 blocks with comments and symbols in
  - Ladder Diagram (LAD),
  - Control System Flowchart (CSF)
  - Statement List (STL)
  - data blocks with comments
  - documentation blocks.

As options, the following can be printed out for each segment:
  - cross references
  - diagnostic setpoint data.

- → ***Block list***
- → ***Assignment list*** sorted according to various criteria

- → ***Reference data*** such as program structure, cross reference list, I/Q/F list or checklist from a program file
- → ***Text files*** (ASCII files)
- → ***Doc commands***, which you can edit and store for each function. The files containing the doc commands can be nested with special calls in the files to provide you with a doc command call structure. These call structures can be displayed graphically.

You can select the printer setting in the → *Settings, Printer parameters* before printing out.

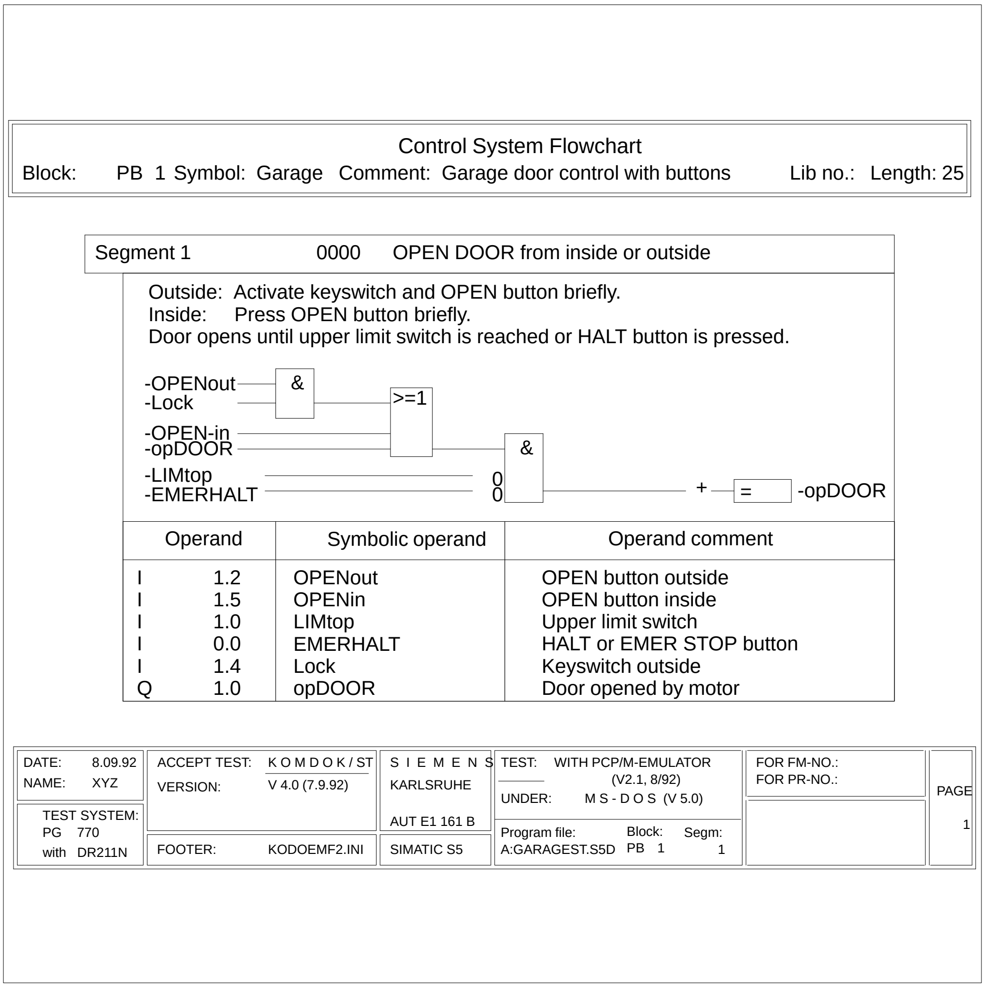
Selecting the Enhanced Functions

When you select the “enhanced output” function, a menu is displayed in which you can select the following elements for output:

- **Program sections** → block lists, blocks, assignment lists
- **Reference data** → program structure, cross references, I/Q/F list, checklist.
- **Text files**
- **Doc commands**

Example of a Printout

The first printout illustrates enhanced output and the second is a standard printout. Note the difference between the two.



Control System Flowchart

Block: PB 1 Symbol: Garage Comment: Garage door control with buttons Lib no.: Length: 25

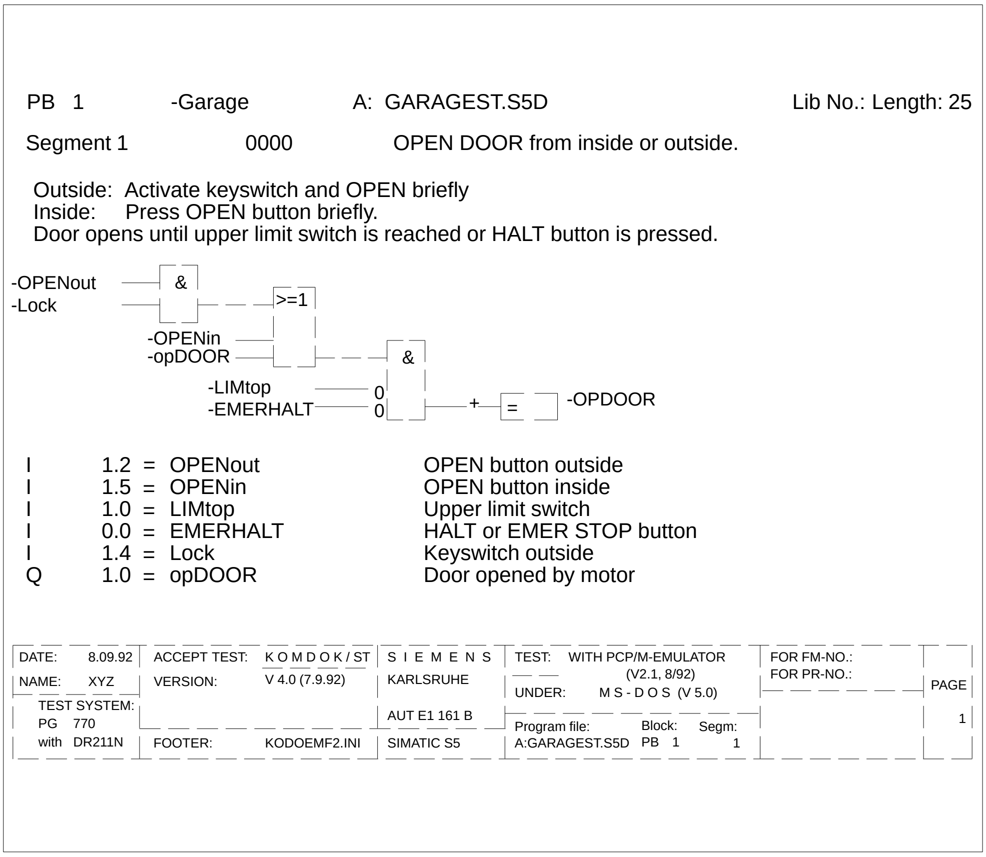
Segment 1 0000 OPEN DOOR from inside or outside

Outside: Activate keyswitch and OPEN button briefly.
Inside: Press OPEN button briefly.
Door opens until upper limit switch is reached or HALT button is pressed.

Operand		Symbolic operand	Operand comment
I	1.2	OPENout	OPEN button outside
I	1.5	OPENin	OPEN button inside
I	1.0	LIMtop	Upper limit switch
I	0.0	EMERHALT	HALT or EMER STOP button
I	1.4	Lock	Keyswitch outside
Q	1.0	opDOOR	Door opened by motor

DATE: 8.09.92 NAME: XYZ	ACCEPT TEST: KOMDOK/ST VERSION: V 4.0 (7.9.92)	SIEMENS KARLSRUHE AUT E1 161 B	TEST: WITH PCP/M-EMULATOR (V2.1, 8/92) UNDER: MS-DOS (V 5.0)	FOR FM-NO.: FOR PR-NO.:	PAGE 1
TEST SYSTEM: PG 770 with DR211N	FOOTER: KODOEMF2.INI	SIMATIC S5	Program file: A:GARAGEST.S5D Block: PB 1 Segm: 1		

Figure 3-73 Enhanced Printout of a Control System Flowchart



PB 1 -Garage A: GARAGEST.S5D Lib No.: Length: 25

Segment 1 0000 OPEN DOOR from inside or outside.

Outside: Activate keyswitch and OPEN briefly
Inside: Press OPEN button briefly.
Door opens until upper limit switch is reached or HALT button is pressed.

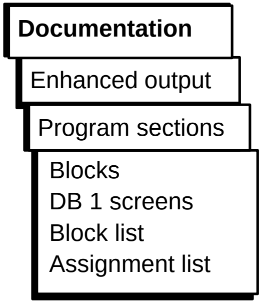
-OPENout
-Lock
&
>=1
-OPENin
-opDOOR
&
-LIMtop
-EMERHALT
0
0
+
=
-OPDOOR

I	1.2	=	OPENout	OPEN button outside
I	1.5	=	OPENin	OPEN button inside
I	1.0	=	LIMtop	Upper limit switch
I	0.0	=	EMERHALT	HALT or EMER STOP button
I	1.4	=	Lock	Keyswitch outside
Q	1.0	=	opDOOR	Door opened by motor

DATE: 8.09.92	ACCEPT TEST: K O M D O K / ST	S I E M E N S	TEST: WITH PCP/M-EMULATOR (V2.1, 8/92)	FOR FM-NO.:	
NAME: XYZ	VERSION: V 4.0 (7.9.92)	KARLSRUHE	UNDER: M S - D O S (V 5.0)	FOR PR-NO.:	PAGE
TEST SYSTEM: PG 770		AUT E1 161 B	Program file: Block: Segm:		1
with DR211N	FOOTER: KODOEMF2.INI	SIMATIC S5	A:GARAGEST.S5D PB 1 1		

Figure 3-74 Simple Printout of a Control System Flowchart

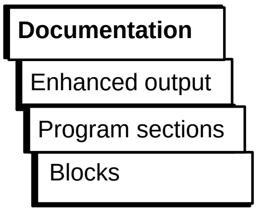
Program Sections



A menu is displayed, in which you can activate the output of the following data:

→ Blocks
→ DB1 screens
→ Block lists
→ Assignment lists

Blocks



This function prints out blocks in the LAD, CSF or STL methods of representation with or without references, with or without diagnostic setpoint data, in A3 or A4 format. You can also direct the printout to a file (*LS.INI).

Settings

The following must be set:

- program file
- footer file (A3)
- symbols file (only when “symbols:yes” is set)
- XRF file (→ *Generate XRF*).
- printer file (the defaults apply to the PT88)
- method of representation

For more information about settings, refer to → *Project*.

Operation

The job box “Documentation of blocks” is displayed. Here, you can make your selections (→ *Graphical user interface, Job box*).

In the following table only the inputs specific to this function are explained.

Input	Explanations
Option	
None	
With forward and backward refs.	**Forward references**: If operands are assigned in the printed segment, the program sections are also printed out in which the scans occur. **Backward references:** If outputs or flags are scanned in the printed segment, the program sections are printed out in which the assignments occur. A line in the printout contains as many cross references per statement as permitted by the layout. The characters “>>>” at the end of the line indicate that there are further cross references in the program.
With diagnostic setpoint data	
Layout	
Standard	If you press ***SHIFT F8*** an example of a standard format is displayed.
Optional	Only relevant in A3 format for the CSF method of representation and for the data block list. The printout is similar to “standard output” in A4 format (left CSF. right cross references).

DB1 Screens

Documentation
Enhanced output
Program sections
DB1 screens

This function prints out the data block with the I/O assignment in A3 or A4 format. You can also output to a file (*LS.INI)

The following must be set:

- program file
- footer file (A3)
- printer file

For more information about settings, refer to → *Project*.

Operation

The job box “DB1 Screens” is displayed. Here, you can make your selections (→ *Graphical user interface, Job box*).

Block List

Documentation

Enhanced output

Program sections

Block list

With this function, you can output a block list in A3 or A4 format on paper or to a file (*LS.INI).
The list contains all the program and data blocks of the selected program file.

You obtain the following information about the listed blocks:

- block type
- block number
- symbolic identifier (if you selected “symbols: yes”)
- operand comments
- block length
- LIB number
- documentation files with length information
- footer

Settings

The following files must also be set:

- program file
- symbols file (only when “symbols:yes” is set)
- footer file (A3)
- printer file (the defaults apply to the PT89)

For more information about settings, refer to → *Project*.

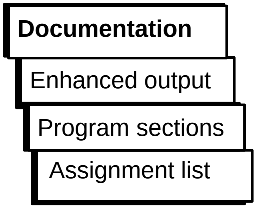
Operation

Depending on the setting, a block list is printed out or output to the selected file. While the block list is being generated, the message

```
printout block list
```

is displayed. When this message disappears and if no error message is displayed the function is complete and the block list is output.

Assignment List



Documentation

Enhanced output

Program sections

Assignment list

You can output an assignment list as follows:

– in sequential form, as edited
– sorted according to absolute operands
– sorted according to symbolic operands.

The following files must also be set:

– symbols file
– footer file (A 3)
– printer file (the defaults apply to the PT89)

For more information about settings, refer to → *Project*.

Operation

The job box “Documentation assignment list” is displayed. Here, you make your selections (→ *Graphical user interface*, *Job box*).

You can output the assignment list in the following modes:

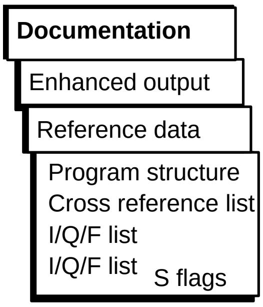
Inputs	Explanations
Option	
Standard output of the seq file	Unsorted output. The symbols setting is not relevant.
Seq file sorted according to absolute operands	The output is sorted according to absolute operands. A new page is started for each of these operands which are output in the order I, Q, F, S, T, C, B, P, D. “Symbols: yes” must be set.
Seq file sorted according to symbolic operands	The output is sorted according to symbolic operands. A new page is started for each of these operands which are output in the order I, Q, F, S, T, C, B, P, D. “Symbols: yes” must be set.
Layout	
Standard	If you press ***SHIFT F8*** or the ***Help*** key, an example of a standard format is displayed.
Optional	Only relevant in A3 format. Operation as described above.

As soon as you exit the job box with ***OK***, the following message flashes on the screen:

```
Printout assignment list
```

If this message disappears, the function is completed and, providing no error message has occurred, the assignment list is output.

Reference Data



A menu is displayed, in which you can activate the output of the following list and data:

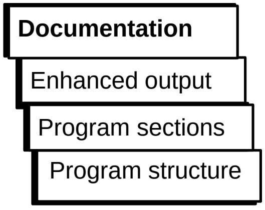
→ *Program structure*

→ *Cross reference list*

→ *I/Q/F list*

→ *Checklist*

Program Structure



This function outputs the block calls in a program file in A3 or A4 format on paper or to a file (*LS.INI). The output has the following conventions:

- The type of block call is specified before each block
- The block name is entered

  in **absolute** form

  and in **symbolic** form (only when you have selected “symbols: yes” → *Project*).
- The maximum nesting depth that can be recorded is 9.
- You can output with or without data blocks.

The following calls are listed:

JU	Unconditional block call
DOU	Unconditional function block (FX) call
JC	Conditional block call
DOC	Conditional function block (FX) call
C	Data block call
CX	Data block (DX) call
G	Generate data block
GX	Generate data block (DX)
AI	Block as parameter (call formal operand)
#	Block call
REC	Recursive block call

Example of a Printout

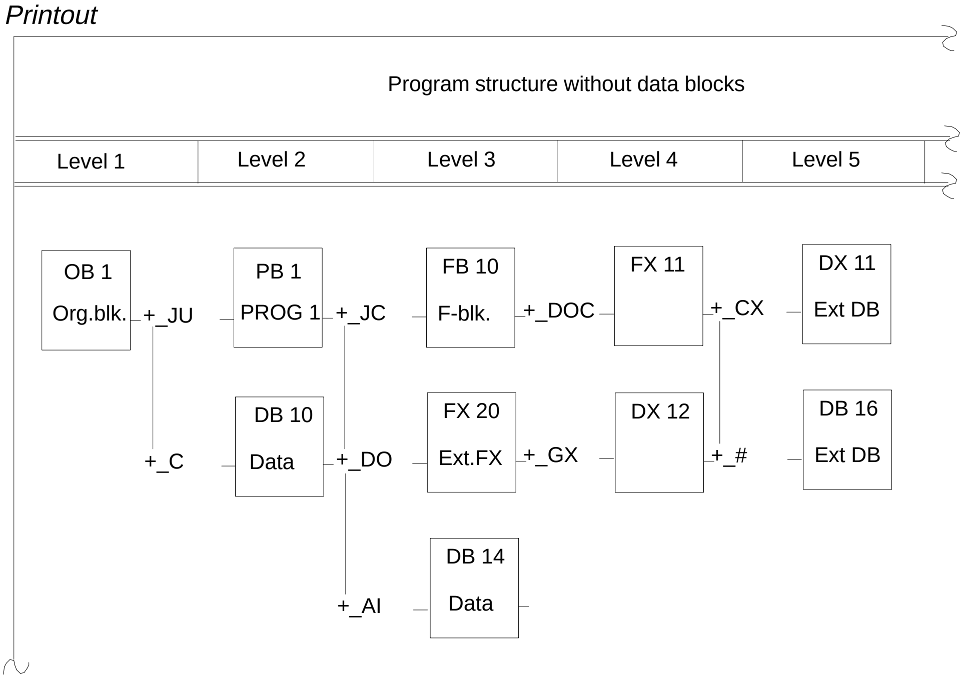


Figure 3-75 Output of a Program Structure without DB

Settings

The following files must also be set:

- program file
- symbols file (only when “symbols:yes” is set)
- XRF file (→ *Management, Generate XRF*)
- footer file (A3)
- printer file (the defaults apply to the PT89)

For more information about settings, refer to → *Project*.

Operation

The job box “Documentation of program structure” is displayed (→ *Graphical user interface*).

Input field	Explanations
Program file	Cannot be selected here. Must be preset (→ *Project*)
Structure from block	The structure of the program is output starting from the specified block.
without DB calls	Data blocks are ignored in the structure.
with DB calls	Data blocks are included in the structure.
Output	As in all job boxes.

As soon as you exit the job box with ***OK***, the following message flashes on the screen:

```
Printout program structure
```

When this message disappears and if no error message has occurred, the function is completed and the program structure output.

Cross Reference List

Documentation

Enhanced output

Reference data

Cross reference list

With this function, you output cross references within the program file according to certain criteria from an existing cross reference list (*XR.INI).

The information is compiled:

– cross reference list according to operand IDs, e.g. B, I, Q, F ...

– cross reference list according to single symbolic or absolute operands (e.g. I 1.0, MOTOR) in the preset file.

Note

Make sure there is always an up to date cross reference list (XRF file) of the valid program file when outputting cross references (→ *Management*, → *Generate XRF*).

If you modify the program, the cross reference list must be regenerated.

Settings

The following files must also be set:

- program file
- symbols file (only when “symbols:yes” is set)
- XRF file
- footer file
- printer file (the defaults apply to the PT88)

Operation

The job box “Print XRF list” is displayed. Here, you can make your selections (→ *Graphical user interface, Job box*).

Input field	Explanations
Selection all elements	All the elements are output in the order I, Q, F, S, T, C, B, P, D, each type on a separate page.
Flags, extended flags, data block, inputs, timers, I/Os, outputs, counters, block calls	These operands are selected singly. A cross reference list is then only output for these operands.
Single operand	Specify a single operand (absolute or symbolic). ***F3*** = *Select* is not possible here. ***SHIFT F8*** provides information.
Layout Standard	If you press ***SHIFT F8*** or the ***Help*** key, an example of a standard format is displayed.
Optional	Only relevant in A3 format. Operation as above.
Standard in compact form	Compact means: if an operand in a segment is addressed n times with the same operation, the segment is not listed n times but only once.

As soon as you exit the job box with ***OK***, the following message flashes on the screen:

```
Printout XRF list
```

When this message disappears and if no error message has occurred, the cross reference list is output.

I/Q/F List

Documentation

Enhanced output

Reference data

I/Q/F list

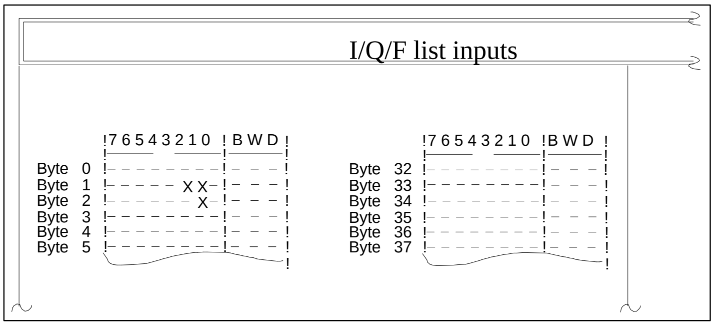
With this function, you output an I/Q/F list. The I/Q/F list takes the form of a table and provides you with an overview of which bit is occupied in the I, F, Q operand areas. One line is reserved for each byte of an operand area, in which the 8 possible bits are marked. In addition, the I/Q/F list also indicates whether the command processes

– a byte (**B**)
– a word (**W**)
– a double word (**D**)

Meaning of the identifiers for bits and bytes in an I/Q/F list:

Identifier	Explanation
”Blank”	The operand is addressed as a byte, word or double word operation and not as a bit operation.
–	The operand is not addressed.
X	A bit operation is performed on the operand.
#	The operand follows DO FW or DO DW operation.
’ ’	The operand is addressed as a byte, word or data word operation, not as a bit operation.
S	The operand is addressed in a standard function block.
?	The operand occurs as a parameter for an FB call.
!	The operand is addressed in a standard FB and in a user FB.

Example of I/Q/F List



```
                          I/Q/F list inputs

          !7 6 5 4 3 2 1 0 ! B W D !             !7 6 5 4 3 2 1 0 !B W D !
          !                !       !             !                !       !
Byte   0  !- - - - - - - - !- - -  !  Byte  32   !- - - - - - - - !- - -  !
Byte   1  !- - - - - - X X-!- - -  !  Byte  33   !- - - - - - - - !- - -  !
Byte   2  !- - - - - - - X-!- - -  !  Byte  34   !- - - - - - - - !- - -  !
Byte   3  !- - - - - - - - !- - -  !  Byte  35   !- - - - - - - - !- - -  !
Byte   4  !- - - - - - - - !- - -  !  Byte  36   !- - - - - - - - !- - -  !
Byte   5  !- - - - - - - - !- - -  !  Byte  37   !- - - - - - - - !- - -  !
                                   !                                      !
```

Figure 3-76 I/Q/F List of the Inputs

Settings

The following files must also be set:

- program file
- XRF file (→ *Management, Generate XRF*)
- footer file (A3)
- printer file (the defaults apply to the PT88)

For more information about settings, refer to → *Project*.

Operation

An I/Q/F list is printed out or output to a file without STEP 5 requiring further information. During the output of the I/Q/F list, the following message is displayed inversely on the screen:

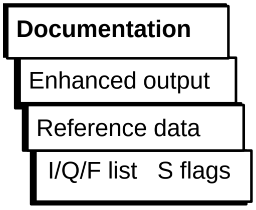
```
Printout I/Q/F list
```

When this message disappears and if no error has occurred, the function is completed and the I/Q/F list output.

Note

Make sure there is always an up to date cross reference list (XRF file) of the valid program file when outputting cross references (→ *Management* → *Generate XRF*).

I/Q/F List



This function outputs the I/Q/F list for the S flags (see Fig. 3-76 “I/Q/F list”).

Checklist

Documentation
Enhanced output
Reference data
Checklist

This function searches through the program file. Depending on the option selected, the following information is output:

Object	Explanation
Free operands	These are operands that occur in the assignment list but not in the program blocks output in the order I, Q, F, S, T, C, B, P, D.
No symbol	These are operands in the program blocks to which no symbol is assigned in the assignment list. These operands are output in ascending order.
Setpoint data absent I/Q/F operands	These exist in the diagnostic data record but there are no setpoint data assigned to them.

Settings

The following files must also be set:

- program file
- symbols file (→ *Management, Generate XRF*)
- XRF file
- footer file
- printer file (the defaults apply to the PT88)

For more information about settings, refer to → *Project*.

Operation

The job box “Documentation checklist” is displayed. Here, you can make your selections (→ *Graphical user interface, Job box*).

Text Files

Documentation
Enhanced output
Text files

With this function you can print out **LS files** or ASCII files or output them to an LS.INI file. Text files can have footers added to them although this is not part of the text file itself. You can therefore add a footer later.

Settings

The following must also be set:

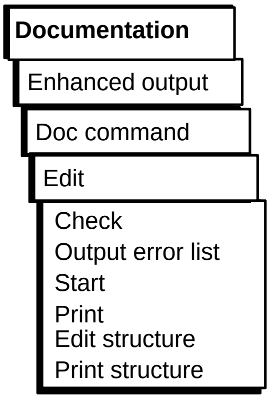
- footer file (only if a footer is required)
- printer file (the defaults apply to the PT89)

For more information about settings, refer to → *Project*.

Operation

The job box “Print ASCII file” is displayed. Here, you can make your selections (→ *Graphical user interface, Job box*).

3.6.3 Doc Commands



You can execute all the functions of the enhanced output using doc commands. These doc commands are put together like a program in a file (submit file) and can be executed by calling this file. The way in which you use the doc commands decides on the type and order of output.

The following functions are available to process doc commands:

→ *Editing doc commands*

→ *Checking doc commands* This triggers a test which checks whether all the commands can be executed. If errors are detected, they are written to an error list.

→ *Executing doc commands*

→ *Outputting doc commands* on the printer or to a file

→ *Outputting the error list* You can output the list of errors if errors were detected during the test.

A doc command string consists of doc commands for

- presets (**$**)
- commands (**–**)
- comments (**;**) (if required)

Structure of the Doc Commands

You can call individual doc command files by means of a suitable statement in a doc command sequence (Fig. 3-79). Following the call, the doc commands in the opened file are executed. Once the sequence of doc commands has been executed, the invoking doc command sequence is continued.

With these commands, you can create a series of statements (structures). To allow a better overview of possibly complex structures, the two following functions are available:

→ *Editing the structure*

The combination of individual doc command files is represented graphically.

→ *Print out the structure*

Example

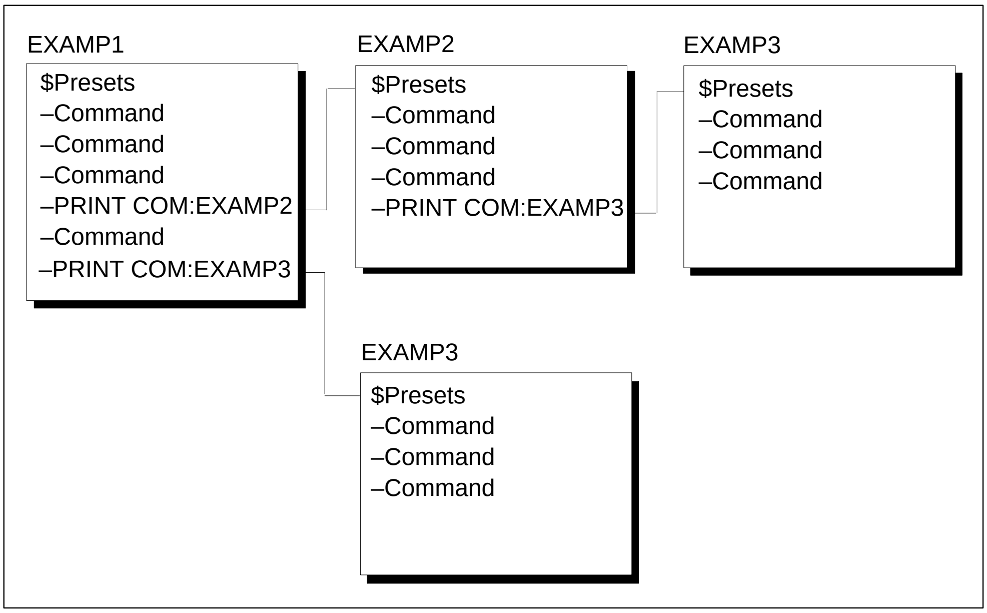


Figure 3-77 Structures of the Doc Commands (Example)

Syntax of the Doc Commands

Presets:

Doc command	Explanation
$LAD, $CSF, $STL	**Method of representation:** of the Ladder Diagram (LAD), Control System Flowchart (CSF), Statement List (STL).
$PROG:X:NNNNNN	**Program file:** to select the file in drive X under the name NNNNNNST.S5D.
$SYMB:X:NNNNNN	**Symbols file:** to select this file in drive X under the name NNNNNNZ0.INI.
$SYMB:NO	**Symbolic operands:** are not output.
$FOOT:X:NNNNNN	**Footer file:** selected in drive X under the name NNNNNNF2.INI.
$PRFI:X:NNNNNN	**Printer file:** selected in drive X under the name NNNNNNDR.INI.
$PATH:X:NNNNNN (PATH name)	**Path file:** the files in the path are declared valid.
$PAGE:nnnn	**Page number:** the page number is incremented from the number nnnn.
$PLST:X:NNNNNN	**Output to file:** all outputs are stored on drive X under the file name NNNNNNLS.INI
$PLST:NO	**Output to printer again**.
$CHARSET:ASCII	**Layout:** use the ASCII character set (broken lines).
$CHARSET:CHA. GRAPHICS	**Layout:** use the IBM character set.
$DIR	**Directory:** from this doc command onwards, a directory is kept. This preset can no longer be reset in the active submit.
$PAUSE:COMMENT	**Interrupt** processing the doc command. The comment is displayed at the lower edge of the screen. By pressing a key the pause is terminated.

Commands

Blocks

Doc commands	Explanation
–BLOCK:A	All blocks
–BLOCK:OB	All organization blocks
–BLOCK:PB	All program blocks
–BLOCK:FB	All function blocks
–BLOCK:FX	Extended function blocks
–BLOCK:SB	All sequence blocks
–BLOCK:DB	All data blocks
–BLOCK:DX	Extended data blocks
–BLOCK: **(e.g. PB1, PB2–PBn)**	A list of blocks
–BLOCK:PBx, 1, 3–5	A list of single segments of a block.

If blocks are output with cross references or diagnostic setpoint data, you must indicate this with an option.

Blocks

Doc commands	Explanation
–BLOCK(R):A	All blocks with cross references.
–BLOCK(O):PBx	PBx in an optional layout (only relevant for CSF and A3 output).
–BLOCK(RO):PBx	PBx with cross references in an optional layout (only relevant for CSF and A3 output).
–BLOCK(D):PBx	PBx in the preset method of representation (LAD, CSF, STL) with diagnostic setpoint data.
–BLOCK:#NNNNNN	Documentation block with the name NNNNNN (max. 8 characters).
–BLOCK:PBSO.n	Setpoint data of PBn.

Block List

Doc commands	Explanation
–BLST	Output the block list of the preset program file.

I/Q/F List

Doc commands	Explanation
–BLOCK(R):PB	All program blocks with cross references.

Nested doc Command

You can call a doc command sequence from other doc command sequences. The maximum nesting depth is 6. Recursive calls are not allowed and are rejected during the test or when a doc command file is started (→ *edit structure*).

Doc commands	Explanation
–DOC–COM:x:nnnnnn	The doc command file nnnnnnSU.INI is called and started.

Directory

A directory of all previous printouts is output if you activate the preset “$DIR”.

Doc commands	Explanation
–DIR	The directory is output with the current footer. The page number begins automatically at I and is restored on completion of the directory.
–DIR:n	The page numbering of the specified directory begins at n (n= 1, 2...)

Checklist

Doc commands	Explanation
–CHECKLIST/FO	The operands occurring in the assignment list but not in the blocks are listed.
–CHECKLIST/NS	The operands used in the blocks but without a symbol in the assignment list are listed
–CHECKLIST/SA	With the SP data checklist, you can list I/Q/F operands defined as an “assignment” but to which no SP data were assigned.

Program Structure

Doc command	Explanation
–XRF:program, (OBn)	Output the program structure from OBn (n=0–255), without data blocks.
–XRF(D):program, (PBn)	Output the program structure from PBn (n=0-255), with data blocks

XRF List

Doc commands	Explanation
–XRF:GENERATE	The reference list (*XR.INI) of the set program file is generated.
–XRF:PRINTOUT, (I)	Output the input operands.
–XRF:PRINTOUT, (Q)	Output the output operands.
–XRF:PRINTOUT, (F)	Output the flags.
–XRF:PRINTOUT, (S)	Output all S flags.
–XRF:PRINTOUT, (T)	Output all timers.
–XRF:PRINTOUT, (C)	Output all counters.
–XRF:PRINTOUT, (B)	Output all blocks.
–XRF:PRINTOUT, (P)	Output all I/Os.
–XRF:PRINTOUT, (D)	Output all data.
–XRF:PRINTOUT, (X)	Collective command for all elements.
–XRF:PRINTOUT, (I1.n)	Output the XRF list of an absolute operand (n = 0 – 7).
–XRF:PRINTOUT, (–SYMBOL)	Output the XRF list of a symbolic operand (e.g. -SYMBOL).
–XRF(C):PRINTOUT, (I)	Output the XRF list of an input operand in compact form. If the input is used more than once in a segment
–XRF(O):PRINTOUT, (Q)	The optional form of the XRF list is output. In contrast to the standard the cross references are not output sorted according to blocks but according to **operations, blocks and segments**.

I/Q/F List

Doc command	Explanation
–XRF:IQF	Output the I/Q/F list. The XRF list must exist. (→ *Management, Generate XRF)*
–XRF:IQF **S FLAGS**	Output the I/Q/F list of the S flags.

Assignment List

Doc command	Explanation
–SYMF:SEQ	Output the source (sequential) file (unsorted).
–SYMF:SYM	Output sorted acc. to symbolic operands.
–SYMF:ABS	Output sorted acc. to absolute operands.
–SYMF(O):SEQ	Output unsorted single column (only relevant in A3 format).

Editing Doc Commands

Documentation

Enhanced output

Doc commands

Edit

To edit doc commands, you can activate auxiliary functions using the function keys. The edited statements are stored in a submit file (*SU.INI).

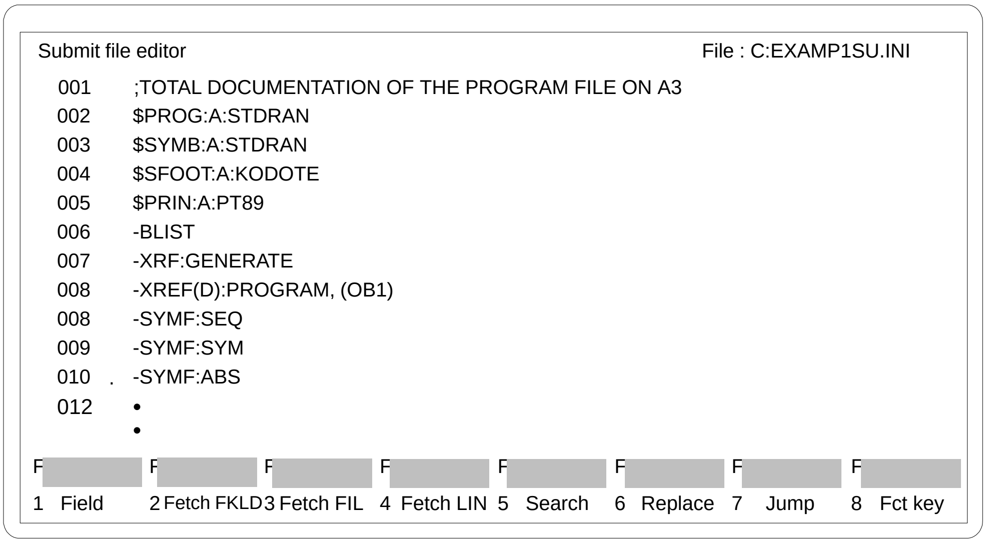
Apart from this fixed function key assignment, you can also assign texts or commands to function keys which you activate with ***SHIFT F1*** to ***SHIFT F7***.

Operation

A job box is displayed in which you select a submit file (→ *Graphical user interface, Job box)*. The new file name is entered in the settings box (→ *Project, Settings, Page 1*). As soon as the screen below is displayed, the cursor is positioned in the first editing line. You can now edit.

Example

Screen display.



```
Submit file editor                                          File : C:EXAMP1SU.INI
   001     ;TOTAL DOCUMENTATION OF THE PROGRAM FILE ON A3
   002     $PROG:A:STDRAN
   003     $SYMB:A:STDRAN
   004     $SFOOT:A:KODOTE
   005     $PRIN:A:PT89
   006     -BLIST
   007     -XRF:GENERATE
   008     -XREF(D):PROGRAM, (OB1)
   008     -SYMF:SEQ
   009     -SYMF:SYM
   010  .  -SYMF:ABS
   012     •
           •
```

Figure 3-78 Submit File Editor

Function Key Assignment

The following section and table describes the key strokes to assign functions to keys.

F1 (Key level 1)	Enter the 1st field delimiter. Change to the 2nd key level.
F3 (Key level 2)	Select the file name for storing the field. Change to the 3rd key level.
F6 (Key level 3)	The field is stored under the selected file name.

The following table shows the effects of the function keys and which key combinations are possible.

Key level			**Effect of the function keys**
1	**2**	**3**	
F1			Store the input with the ***Insert*** key. ***Cursor keys*** → *Appendix A4, key assignment.* Field The 1st field delimiter is marked in the current line with <B>. The 2nd field delimiter can be moved over further lines with the cursor keys.
	F1		Field The marked field is stored for the current session.
	F3		File The field is stored under a selectable file name but it remains in the buffer.
		F6	Enter The field is stored in the selected file.
		F8	Return Return to previous key level without action.
	F4		Find (text) Search for a max. 30 character string in a field. If the text is found, the 2nd field delimiter is set in this line.
		F5	Repeat Repeat the last search.

Key level			**Effect of the function keys**
1	**2**	**3**	
		F6	Srch fwd Text searched for towards the end of the file.
		F7	Srch back Text searched for towards the start of the file.
		F8	Return Return to previous key level without action.
	F6		Enter The block is stored for the current session.
	F7		Jump Jump to the start/end of the file or to a selectable line number.
		F6	To start Jump to the start of the file.
		F7	Line Jump to the selected line.
		F8	End Jump to the end of the file
	F8		Return Return to previous key level without action.
F2			Fetch fld The currently buffered field is fetched and inserted after the cursor position.

Table 3-9 Existing Submit File

Key level			Effect of the function keys
1	2	3	
F3			Fetch file A selectable submit file is fetched from a selectable drive.
	F1		File The file is fetched.
	F2		Fct keys The function assignment is fetched from the file and is active from now on.
	F6		Enter The file is fetched without function assignment.
	F8		Return Return to previous key level without action.

Table 3-10 Fetch Line/Find Text

Key level			Effect of the function keys
1	2	3	
F4			Fetch line A previously deleted line (with the delete key) is fetched back → **Key assignment**.
F5			Find Search for a max. 30 character string. The repetition factor can be selected.
	F5		Repeat Repeat the last search
	F6		Srch fwd Search towards the end of the file.
	F7		Srch back Search towards the start of the file.
	F8		Return to previous key level without action.

Table 3-11 Replace Character String

Key level			Effect of the function keys
1	**2**	**3**	
F6			Replace A character string is replaced by another. You can enter a max. 30 character long string and a repetition factor. The text is replaced by the second.
	F1		Rep? fwd Search to end of file. Replacement must be confirmed.
		F1	Yes - The text is replaced.
		F2	No - The text is not replaced.
		F8	Return Return to previous key level without action.
	F2		Rep? back Search to start of file. Replacement must be confirmed.
		F1	Yes - The text is replaced.
		F3	No - The text is not replaced.
		F8	Return Return to previous key level without action.
	F3		Rep fwd Search to end of file. Text replaced without conf.
	F4		Rep back Search to start of file. Text replaced without conf.
	F6		Repeat Repeat the last replacement.
	F8		Return Return to previous key level without action.

Table 3-12 Jump

Key level			Effect of the function keys
1	2	3	
F7			Jump Jump to the start/end of the file or to a selectable line number.
	F6		Start Jump to the start of the file.
	F7		Line Jump to the selected line.
	F8		End Jump to the end of the file.

Table 3-13 Editing Function Keys

Key level			Effect of the function keys
1	2	3	
F8			Fct keys Assigns the keys SHIFT F1-F8 with a selectable max. 30 character string. This string is entered in the line marked by the cursor in the editing mode when the function key is pressed (SHIFT F1 - F8). e.g. SHIFT+F1: |$PROG:C:FILE\| SHIFT+F2: \|$SYMB:C:SYMDAT|
	F4		Fetch 1 Fetch back the characters deleted with the delete key(→ **Key assignment**).
	F6		Enter The function key assignment is entered.
	F8		Return The function key assignment is entered.

Checking Doc Commands

Documentation

Enhanced output

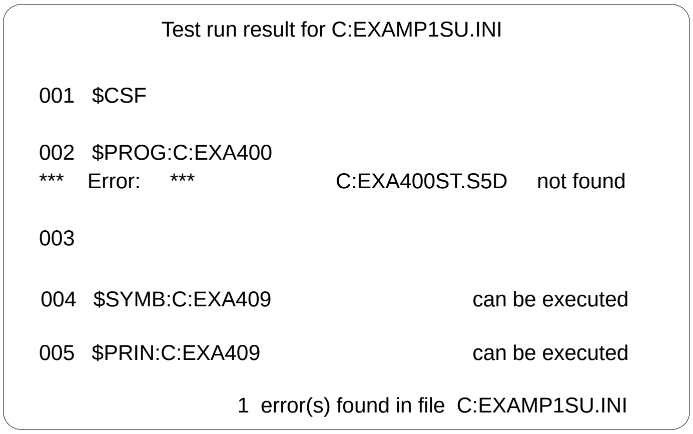
Doc commands

Check

The feasibility of doc commands is checked in a selectable file. If errors are recognized, the cause of the errors is entered in an *SF.INI file.

Example of an Error Message

The following figure illustrates the error messages that can be displayed.



```
                    Test run result for C:EXAMP1SU.INI

001   $CSF

002   $PROG:C:EXA400
***    Error:    ***                C:EXA400ST.S5D     not found

003

004   $SYMB:C:EXA409                             can be executed

005   $PRIN:C:EXA409                             can be executed

                        1  error(s) found in file  C:EXAMP1SU.INI
```

Operation

The job box “Check doc command file” is displayed. Here, you enter the name of the file you want to check. The check is started as soon as you click on ***OK***.

Note

If no errors are found, no error file is created.

Outputting The Error List

Documentation
Enhanced output
Doc commands
Output error list

Errors found in the functions “Check doc command” or “Execute doc command” are written to a file. You can output these files with this function.

The following must also be set:

- printer file (the defaults apply to the PT89)

For more information about settings, refer to → *Project*.

Operation

The job box “Output error list” is displayed. Here, you can make your selections (→ *Graphical user interface*). The name of the generated error file is set here.

Destination	Explanation
Error file	Name of the error file. The generated error file name is the default. You can select a different name with ***F3***.
Screen	Output directly on the screen.
Printer	Output directly on the printer according to the selections made for printer parameters.
File	Output to a selectable file.

Executing Doc Commands

Documentation
Enhanced output
Doc commands
Start

With this function, you can activate the doc commands in your file.

The current settings remain valid unless you change them with a presets statement ($PROG:...$CSF, etc.). The preset statements are, however, only valid for the time when the doc commands are executed.

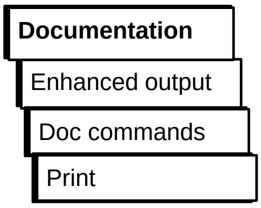
Operation

The job box “Start doc command file” is displayed. Here, you enter the name of the file whose doc commands you want to use in the field “File name”. You can select a file by pressing ***F3*** (→ *Graphical user interface, Job box*). Once you confirm the “Start doc command” with ***OK*** the doc commands are processed.

Note

If errors occur, you can branch to an error list.

Printing Doc Commands



You can print out the content of a doc command file.

Operation

The job box “Documentation of doc command file” is displayed. Here, you enter the name of the file you want to print in the “File name” field. You can select a file with ***F3*** (→ *Graphical user interface, Job box*) *BI*. When you click on ***OK*** the doc commands are printed out.

Editing the Structure

Documentation

Enhanced output

Doc commands

Edit structure

Within the doc commands you can include statements (→ *Structure statements*) which call and start other doc command files. This function shows you how the various doc command files are connected by the structure statements.

This function also allows you to start the doc command editor and modify the statements of the current doc command file.

Example

The figure shows how the editor represents the connections between doc command files established by doc commands.

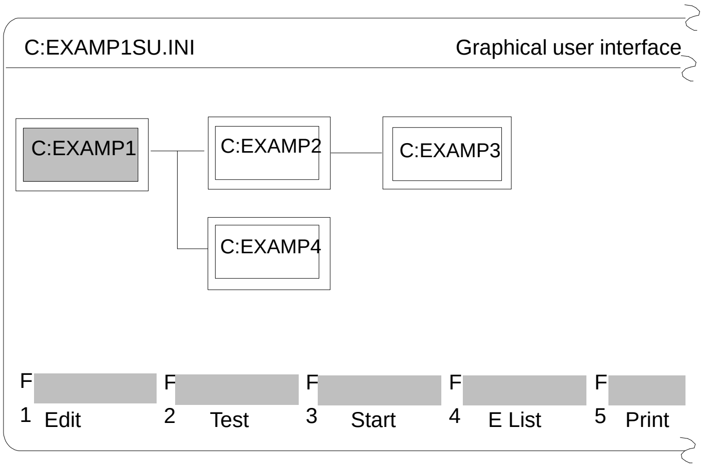


Figure 3-79 Interconnecting Doc Command Files

Operation

The job box “Edit doc command structure” is displayed. Here, you specify a doc command file name or select a name with ***F3***. Using this file as the starting point, the relationship between the doc command files is displayed.

Once you exit the job box with ***OK*** a structure diagram is displayed. The doc command file with which you called the editor is highlighted.

Moving the Marker

You can change the marking of the individual doc command files in the structure display with the ***cursor*** keys (→ *Appendix A4, Key assignment*).

Function key Assignment

The following section explains the significance of the various function keys.

F6 = Key level 1	You want to search for a particular doc command file in the structure file. You change to the 2nd key level.
F1 = Key level 2	The first structure statement file is marked.

The table shows which key combinations are possible and the effects of the function keys.

Key level		**Effect of the function keys**
1	**2**	
F1		Edit The doc command editor is called and the content of the current doc command file is displayed. You can edit these doc commands (→ *Editing doc commands*).
F2		Test The doc command file highlighted (color/gray background) in the structure display is tested. The result is displayed on the screen immediately. If errors are found, they are written to an error file.
F3		Start The doc command file highlighted in the structure display is started. If errors occur during execution, they are written to an error file and displayed on the screen.
F4		E list The error list of the doc command file marked on the screen is displayed and, if required, printed out.

Key level		Effect of the function keys
1	**2**	
F5		Print The doc command file marked in the structure display is output on the printer or to a file depending on the settings.
F6		Search Switch to the search functions.
	F1	To start The first doc command file in the structure display is marked and is now the current file.
	F2	End The last doc command file in the structure display is marked and is now the current file.
	F3	Caller The doc command file via which the structure display was called is marked and is now the current file.
	F4	Error Starting from the currently marked file.
	F6	Srch fwd A selected doc command file is searched for towards the end of the display. If it is found it is marked and is now the current file.
	F7	Srch back A selected doc command file is searched for towards the start of the display. If it is found it is marked and is now the current file.
F8		Return Return to the calling level.
SHIFT F8		Help

Printing the Structure

Documentation
Enhanced output
Doc commands
Print

The structure of connected doc command files is printed out in A3 or A4 format or output to a file (LS.INI).

Settings

The following must be set:

- printer file (the defaults apply to the PT89)
- footer file

For more information about settings, refer to → *Project*.

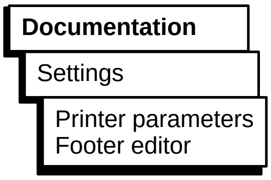
Operation

The job box “Print doc command structure” is displayed. Here, you can make you selections (→ *Graphical user interface*).

Input field	Explanation
Submit file	Name of the doc command file about which you want to see structure information. Starting from this file.
Structure	Here only the structure is displayed.
Structure with doc commands	The content of the doc command files involved is also printed out, each file on a separate page.

3.6.4 Settings

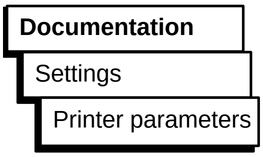
Settings



Before you can print out files or redirect them to a file in a printable format you must do the following:

- Set the parameters for your printer (→ *Printer parameters*).
  A variety of printers can be connected to the programmer. The parameters required for the printer must be set and stored in a printer file (*DR.INI) in the system directory. There are “off the shelf” printer files available for the various printer types. These contain settings for specific printers and the type of printout (portrait, landscape). In the “Settings, page 2” box, you can click “Printer file” to obtain a list of printer files (*DR.INI) available in the system directory. Press ***F3*** to display a printer selection box.
- Select and edit the footer (→ *Footer*).

Setting Printer Parameters



You prepare a control character record for your printer and store it in a file of the type DR.INI. This controls the printout directly on the printer. You make these entries in a selection box.

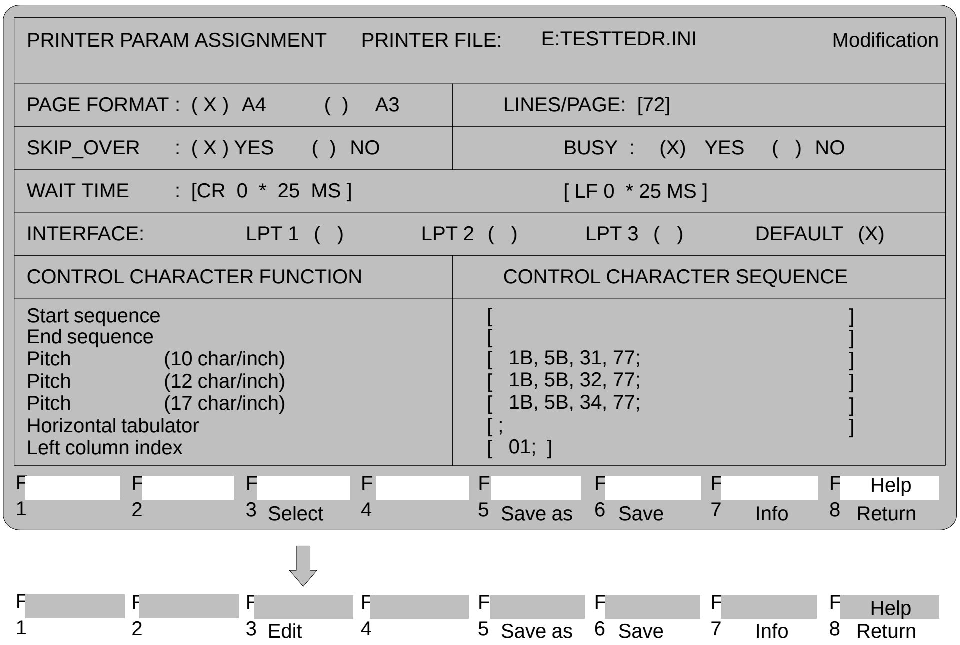
Settings

Select the printer file of type *DR.INI in the “Settings, page 2” box. The asterisk (*) stands for the six-character name of the printer file.

For more information about settings, refer to → *Project*.

Operation

The box “Printer parameter assignment” is displayed (example below). The file C:HP3Q@@DR.INI for the HP III (C) printer was selected in the “Settings, page 2” box.



```
PRINTER PARAM ASSIGNMENT    PRINTER FILE:    E:TESTTEDR.INI              Modification

PAGE FORMAT : ( X )  A4      ( )   A3         LINES/PAGE:  [72]
SKIP_OVER    : ( X ) YES     ( )  NO                BUSY :   (X)   YES    ( )  NO
WAIT TIME    :  [CR  0  *  25  MS ]                 [ LF 0  * 25 MS ]
INTERFACE:              LPT 1  ( )        LPT 2  ( )       LPT 3  ( )       DEFAULT  (X)
CONTROL CHARACTER FUNCTION                    CONTROL CHARACTER SEQUENCE
Start sequence                               [                                  ]
End sequence                                 [                                  ]
Pitch              (10 char/inch)            [   1B, 5B, 31, 77;                ]
Pitch              (12 char/inch)            [   1B, 5B, 32, 77;                ]
Pitch              (17 char/inch)            [   1B, 5B, 34, 77;                ]
Horizontal tabulator                         [ ;                                ]
Left column index                            [   01;  ]
```

Function Keys In this box, you can activate certain functions using function keys as explained in the table.

Key	Function
F3	1. (Select) When the cursor is positioned on an input field in which you can select various parameters, the function key “Select” is displayed. You can select parameters with ***F3***. 2. (Edit) When the cursor is positioned on an input field in which you can type in characters, the function key “Edit” is displayed. You can position the cursor on the character field with ***F3***. 3. (Edit control character functions) When the cursor is on an input field under “CONTROL CHAR FUNCTION”, the “Edit” softkey is also displayed. With ***F3***, you can open an editing window for control characters for your printer. You enter your input with the ***Insert*** key.
F5 = Save as	The PRINTER FILE is stored under the name you select. Once you press this key the cursor jumps to the field with the file name. You can now change this if you wish. The ***Return*** key stores the parameter settings under this name.
F6 = Save	This stores the selected parameters in the current PRINTER FILE:
F7 = Info	With this key, you can obtain information about the field marked by the cursor. You clear this text from the screen using the cursor keys (→ *Appendix A4, key assignment*).
F8 = Return	Return to the calling level.

Parameter Dialog Box

The following list explains entries in the parameter assignment box.

Input field	Explanations
PRINTER FILE	The printer settings are stored in this file. You can specify the name under → *Project* or with "***F5*** (Save as).
PAGE FORMAT	A4 A3
LINES/PAGE	Number of lines per page.
SKIP_OVER: YES NO	The control character FF (form feed) is output to trigger a form feed. The remaining page is output with empty lines up to the number specified in LINES/PAGE providing no lines contain characters.
BUSY	Not relevant for the PT88/PT89/PT10. This only affects older printer types. Following each character sent to the printer, STEP 5 waits a specified time (WAIT TIME) for confirmation from the printer before sending the next character.
NO	No confirmation is expected.
YES	A confirmation is expected.
WAIT TIME	You can set the wait time for a confirmation (in milliseconds).
CR	• for carriage return
LF	• for line feed
INTERFACE	The device interface LPT1, LPT2 and LPT3 on which information is transferred to the printer can be selected by entering an X. The default is LPT1. In the printer files supplied, LPT1 is set (X). The default setting of the PG assigns the parallel device interface to LPT1. No further interfaces for connecting a printer are assigned to the LPT2 and LPT3 ports. **Note** If you change the assignment of the LPT ports (serial printer), remember that with PGs of the 7xx series the COM1 (PLC) and COM2 (mouse) ports are assigned.
CONTROL CHAR FUNCTION	You can edit a control character string for your printer. A character string can be up to a maximum of 127 bytes long. Only hex. characters are allowed.
Start sequence	Before each print job

Input field	Explanations
End sequence	After each print job
Pitch	Here you select the number of characters per inch.
(10 char/inch)	NORMAL
(12 char/inch)	CONDENSED
(17 char/inch)	SUPER CONDENSED
Horizontal tabulator	With this the printer head is positioned on a column. The dummy character for the dynamic entry of this column is “00”. The next column with a printable character is calculated from the current position of the head and the number of blanks following it. This position is entered in the control character string.
Left column index	The dummy character for the horizontal tabulator is calculated with this. It is the index of the left page column of the printer and specifies whether it begins with 0 or 1.

Explanation for the printer names in the system directory.

Name	Meaning
Emul.	Emulation
A3, A4	Page format: A3, A4
Norm.	Print type: normal
Comp.	Print type: compact
L/P	Lines/page
(C)	Identifies printers of other vendors for which Siemens does not guarantee perfect operation.

Footer

You can append a fixed number of automatically generated footer lines to each page of a printout of S5 user programs or S5 program sections. In the “Settings, page 2” box, you can select a footer width of either 80 or 132 characters.

Each footer is stored in its own file and is created using the footer editor (files *F1.INI for 80 characters wide and *F2.INI for 132 characters wide). You can select any combination of S5 files and footers.

Editing Footers

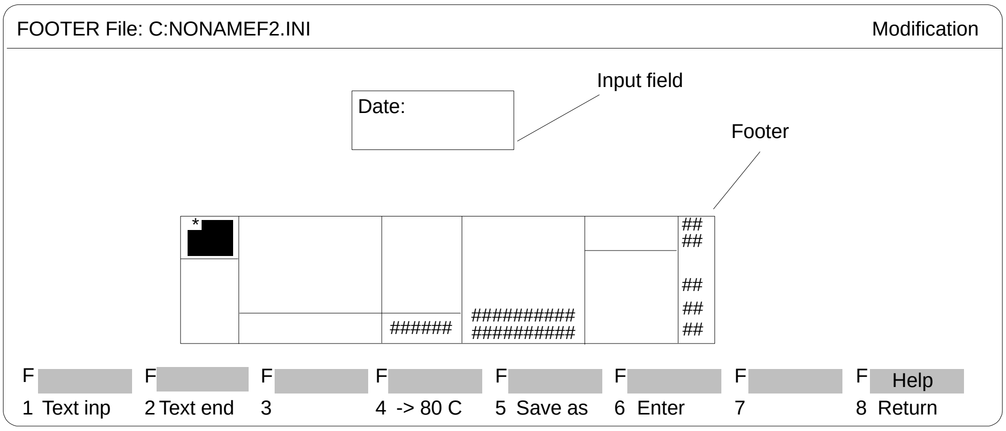
Documentation
Settings
Footer editor

With this function, you can write a new footer or modify an existing one. The size of the editing field displayed is adapted to the number of footer characters. A field in which an entry can be made is highlighted. You cannot write in fields marked with ## since these are reserved for automatically generated text, e.g.

– SIMATIC S5
– Program file
– Block
– Segment
– Page number

Example

The screen displays the editing window for 132 character wide footer. The editing window for an 80 character footer only has 4 fields. The name of the file is displayed at the top left of the screen. The top right of the screen tells you whether it is a new file or whether you are modifying an old one.



FOOTER File: C:NONAMEF2.INI

Modification

Date:

Figure 3-80 Editing Window for 132 Character Wide Footer

Settings

The following files and parameters must be set:

– footer file
– footer width (80 or 132)

For more information about settings, refer to → *Project*.

Operation

When you start the footer function, an editing window is displayed. The upper field is the **input window**. You only have direct access to this field. The lower field is the **footer** in which the text is inserted. When a field in the footer is highlighted, you can enter text for this field in the input window (the cursor flashes in the input window). You can familiarize yourself with the keys relevant for the footer editor in → *Using the footer keys*.

Note

Input field “Date”:
If no date is entered here, it is entered automatically by the system.
When you print using the enhanced mode (KOMDOK), this is overwritten by the current system date.

Using the Footer Keys

With these keys, you can position the cursor and input texts.

Key	Functions
F1	**Activate text input**. The input window is activated and the cursor flashes inside it.
F2	**Deactivate text input**. No input is possible in the input window.
F4	Switch the footer width to 80 or 132 characters.
F5	Change the footer file name. The change can be saved with ***F8*** = ***Return*** and ***YES***.
F6	The edited footer is stored.
F8	Return to the previous level.
SHIFT F8	Return to the previous level.

SHIFT + 4 6 2 8

Cursor in the footer: (SHIFT + a cursor key)

(4) Positions the cursor on the previous left footer field.

(6) Positions the cursor on the next right footer field.

(2) Positions the cursor on the next lower footer field (also without SHIFT).

(8) Positions the cursor on the previous upper footer field 1 (also without SHIFT).

4 6 2 8

Cursor in the input window

(4) Positions the cursor on the character before and replaces it.

(6) Positions the cursor on the next character and replaces it.

(2) Positions the cursor on the next line. If the cursor leaves the input field as a result, text input is terminated.

(8) Positions the cursor on the line above. If the cursor leaves the input field as a result, text input is terminated.

DEL

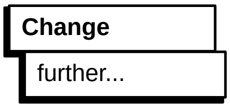
Delete character

The character marked by the cursor is deleted and the following characters brought forward to close the gap.

3.7 Change

With this function you can change to other S5 packages. If they are not already loaded, they must be installed in a directory on a drive. With the “Change” function, you exit the STEP 5 package.

further..



All the installed S5 packages available on the drive and in the directory you have selected are displayed. You can then change to one of these programs.

With the “further” function, you exit STEP 5. The user interface of the selected S5 package is displayed and you can then continue working with the new package.

You can return to STEP 5 from any other S5 package. The STEP 5 settings are retained, so that you can resume work immediately without needing to select new settings.

PG Link

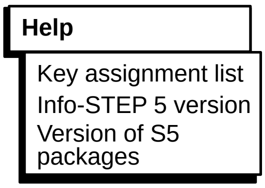
The S5 package “PG Link” is supplied with the STEP 5 package. It is installed in the directory C:\STEP 5\S5_ST\PG_PG. If you set the appropriate path in the selection box, the PG Link program is displayed and you can start it.

Operation

The “Other S5 programs” job box is displayed. Here, the installed S5 packages you can select are displayed. The lower part of the box displays stamp information about the S5 package marked by the cursor.

You make your selection in this box (→ *Graphical user interface*). Once you have selected a package and confirmed your selection with ***OK***, the user interface of the selected package is displayed.

3.8 Help

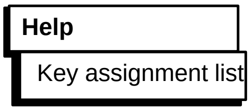


With the functions in this menu, you can obtain information about the currently active STEP 5 package, as follows:

- a list of all the function keys (***F1*** - ***F10*** and ***SHIFT F1*** - ***SHIFT F9***). Using these keys, you can select STEP 5 functions from the main menu directly,
- information about the version of STEP 5 you are currently working with,
- a list of all the program components in the currently active STEP 5 package.

3.8.1 Key Assignment List

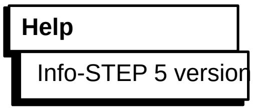
Key Assignment List



This list provides you with information about the function keys you can activate directly in the user interface. These keys allow you to select certain functions directly without using the menus.

When you select this function, a list explaining the functions of the keys is displayed on the screen. You can page through this list.

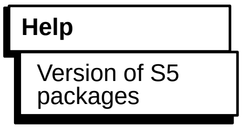
3.8.2 Info-STEP 5 Version



A box is displayed containing information about the currently active STEP 5 package.

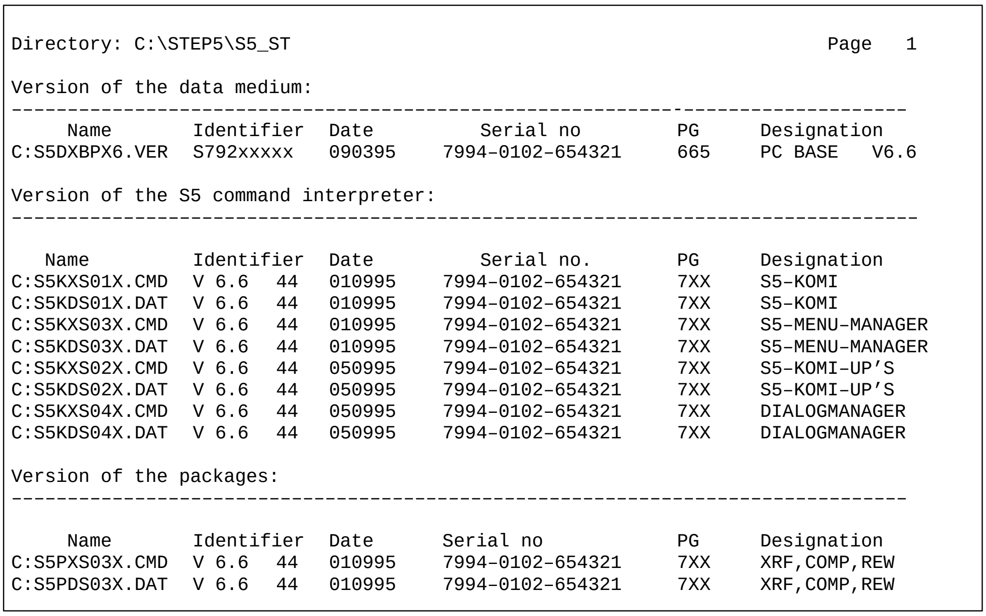
3.8.3 Version of S5 Packages

3.8



A list of all the program components in the currently active STEP 5 package is displayed. You can set the drive and the directory in which the program components are looked for.

The information is output to screen, printer or file. If you output to printer or file, the layout is the same as the standard output.



```
Directory: C:\STEP5\S5_ST                                                  Page   1

Version of the data medium:
--------------------------------------------------------------------------------
     Name       Identifier  Date         Serial no        PG     Designation
C:S5DXBPX6.VER  S792xxxxx   090395    7994–0102–654321    665    PC BASE   V6.6

Version of the S5 command interpreter:
---------------------------------------------------------------------------------

   Name         Identifier  Date         Serial no.       PG     Designation
C:S5KXS01X.CMD  V 6.6  44   010995    7994–0102–654321    7XX    S5–KOMI
C:S5KDS01X.DAT  V 6.6  44   010995    7994–0102–654321    7XX    S5–KOMI
C:S5KXS03X.CMD  V 6.6  44   010995    7994–0102–654321    7XX    S5–MENU–MANAGER
C:S5KDS03X.DAT  V 6.6  44   010995    7994–0102–654321    7XX    S5–MENU–MANAGER
C:S5KXS02X.CMD  V 6.6  44   050995    7994–0102–654321    7XX    S5–KOMI–UP’S
C:S5KDS02X.DAT  V 6.6  44   050995    7994–0102–654321    7XX    S5–KOMI–UP’S
C:S5KXS04X.CMD  V 6.6  44   050995    7994–0102–654321    7XX    DIALOGMANAGER
C:S5KDS04X.DAT  V 6.6  44   050995    7994–0102–654321    7XX    DIALOGMANAGER

Version of the packages:
--------------------------------------------------------------------------------

     Name       Identifier  Date      Serial no           PG     Designation
C:S5PXS03X.CMD  V 6.6  44   010995    7994–0102–654321    7XX    XRF,COMP,REW
C:S5PDS03X.DAT  V 6.6  44   010995    7994–0102–654321    7XX    XRF,COMP,REW
```

Figure 3-81 Example of the Versions of the S5 Packages

Settings

The following must be set:

– footer file (only if footers are selected)
– printer file (for output to printer/file, the default is the PT 88)

For information about making settings, refer to → *Project*.

Operation

The job box for the version of the S5 packages is displayed. Here, you make your selections (→ *Graphical user interface, Job box*).

Directory

The version in the directory displayed here is shown. The standard setting after calling the function is always the S5 system directory. You cannot edit the “Directory” field although the field can be selected with the cursor or mouse. If you select the “Directory” field, you can select the required directory with ***F3*** = *Select* or by double clicking with the mouse.

Description of Technical Resources 4

4.1 S5 Files

This section tells you which directories contain files necessary for STEP 5. For more detailed information about the directories and files on your device refer to the product information.

Under S5-DOS/ST

C:\DOS\

MS-DOS system directory. This contains the MS-DOS operating system.

C:\SIMATIC\S5_ST\

STEP 5 system directory with the STEP 5 basic package and the CP/M emulator, the file S5.BAT, with which you start the STEP 5 basic package and the P-Tools. .
Further files: S5*.CMD, the corresponding S5*.DAT files, the printer files ??????DR.INI and the interface parameter assignment files AS511S0?.DAT.

C:\S5_DATEN\DEFAULT\

This contains the example program which is stored in the directory C:\S5_DATEN\EXAMPLE\ after processing.

C:\S5_DATEN\EXAMPLE\

The example program with the program blocks and assignment list is stored in this directory after processing.

C:\STEP 5\S5_ST\INSTALL

Contains backups of the individual S5 program sections.

C:\STEP 5\S5_ST\PG-PG

Link between two PGs for exchanging STEP 5 blocks and files.

C:\STEP 5\S5_ST\S5_COM

Default directory for optional packages (COMs, PROMs).

4.1.1 STEP 5 Files with Special Functions

This section lists the files in which STEP 5 stores its settings and data. Most of the files are stored in the STEP 5 working directory. The question marks in the file names stand for characters defined by the user.

Paths	**Settings**
S5 MEMORY.DAT	File for the last values entered in the job and selection boxes.
STEP 5 CF.INI	(STEP 5 Configuration File) This contains the path and the name of the ??????PJ.INI file last used. Stored in the STEP 5 system directory.
??????PJ.INI	Data selected in the “Settings” screen form.
Programs	
??????ST.S5D	STEP 5 program file under S5-DOS data management
Assignment list	
??????Z0.SEQ	Sequential, not translated assignment list (S5-DOS data management). Under S5-DOS/MT you can generate and modify this file both under STEP 5 and with the HARD PRO tool SIGNAL.
??????ZF.SEQ	Assignment error list: list of the errors that occurred converting from ??????Z0.SEQ to ??????Z0.INI.
??????Z0.INI	Symbols file (S5-DOS data management), translated assignment list.
??????Z#.INI	Assignment list index files (# = 1 or 2).
??????ZF.INI	Stores the function key assignment.
Printer output	
??????.DR.INI	Printer parameters. Stored in the STEP 5 system directory.
??????F1.INI	Footer file (80 characters)
??????F2.INI	Footer file (132 characters)
??????LS.INI	File to store a redirected printout.
Specific files	
??????XR.INI	(Reference list) XRF file
??????SU.INI	DOC commands (Submit)
??????SF.INI	Submit error list
??????TP.INI	Key macros
Bus selection	
??????AP.INI	Path file The bus paths you edit are stored here. This file is located in the STEP 5 system directory.
??????SD.INI	SYSID file System identification characteristics, e.g. of CPs.

4.2 Data Management S5-DOS

Under the S5-DOS/ST operating system, STEP 5 uses the S5-DOS data management to handle its data.

The S5-DOS data management has been optimized to meet the requirements of STEP 5.

Characteristics:

- Data is saved extremely quickly.
- The data are stored in compressed form. If a certain amount of data has been deleted (more than 10 gaps) the remaining data are reorganized so that the data base is not unnecessarily extended by gaps and so that the data does not need to be broken up on the storage medium. The S5-DOS data management therefore requires little space on the medium.
- The program files are of the type ??????ST.S5D.
- The number of blocks is limited to a maximum of 255. This means that there cannot be a documentation block for every block.
- The assignment list (file of the type ??????Z0.SEQ) for one PLC and one CPU can be created and edited both with the HARDPRO tool SIGNAL and with STEP 5. This must then be converted to a symbols file of the type ??????Z0.INI under STEP 5.

Summary

S5-DOS data management:

Program file for PLC1, CPU1: PRG11@ST.S5D	—	Symbols file for PLC1, CPU1: PRG11@Z0.SEQ (Symbol, Operand, Comment)
Program file for PLC1, CPU2 PRG12@ST.S5D	—	Symbols file for PLC1, CPU2: PRG12@Z0.SEQ
Program file for PLC2, CPU1: PRG21@ST.S5D	—	Symbols file for PLC2, CPU1: PROG21@Z0.SEQ

Appendix A

A.1 Creating the Program for the Example

Creating the elements of a STEP 5 program (program blocks, segments, data blocks, assignment lists) for a given task demands a certain development process. In general, you require the programming instructions for your PLC and should know the basics of the SIMATIC S5 system.

For the simple case of a carwash, the development process is restricted to executing the following steps:

S1: The process to be controlled and the process elements are represented schematically.

S2: The input/output signals are listed and given symbolic names.

S3: The control sequence with its conditions and actions is represented in a decision table according to the verbal description of the process.

S4: The data block is set up.

S5: The blocks of the program are programmed in STL (a segment for each process step).

Step 1:

Schematic representation of the process to be controlled
As preparation before writing the program, the carwash is represented schematically, so that the process peripherals of the controller (sensors/actuators) and their effects in the control sequence can be recognized (Fig. A-1).

To achieve the correct logical combinations in the PLC, it is important to know the way in which the input elements function. When programming, you must know whether the contacts are normally open (NO) or normally closed (NC).

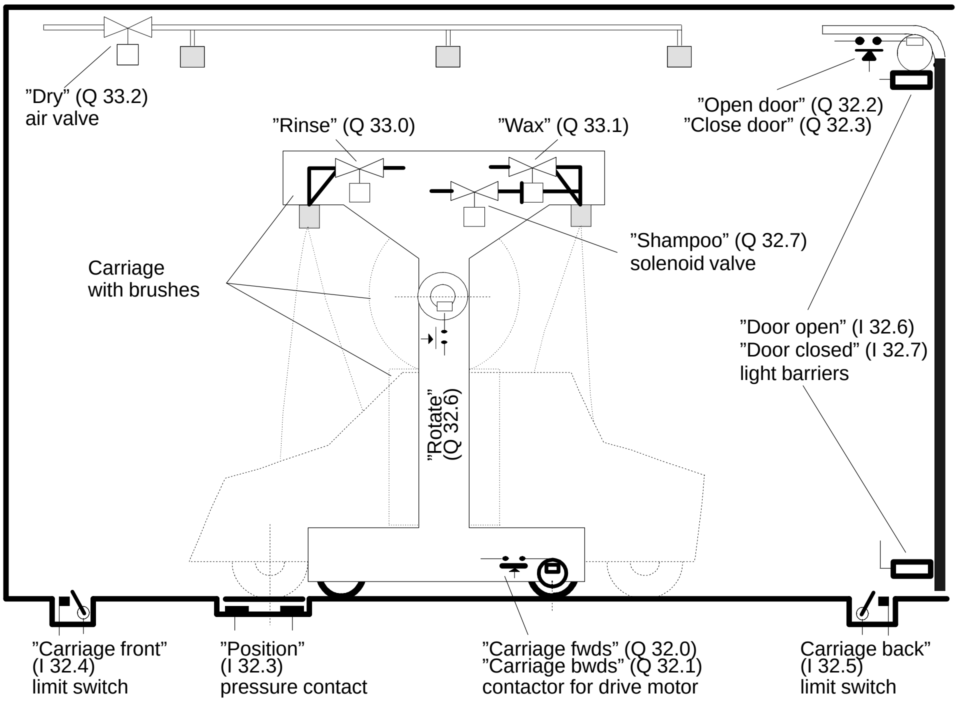


Figure A-1 Carwash with Process Inputs/Outputs

The schematic representation of the carwash provides information for comparing lists of the process inputs/outputs which will be processed by the control system as operands. The process signals for the operation and display elements as shown in Fig. KEIN MERKER must also be added to this list.

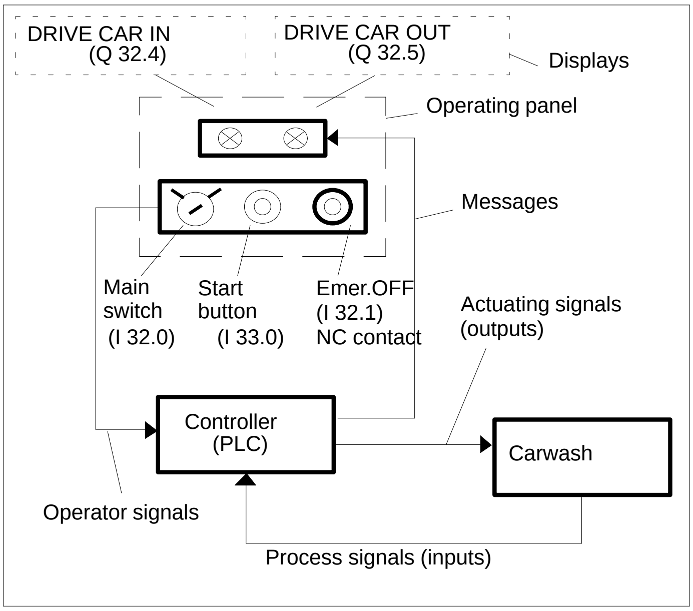


Figure A-2 Control Structure with Operator Inputs/Outputs

All the data transferred to and from the control program via the process interface and required for creating the operand list and describing the process sequences are now known.

Step 2:

Listing the input/output variables

To describe the process and to write the program, it is easier to use symbols for the input/output variables. The plant and operator I/Os are then compiled in a table as shown below.

Table A-1 List of Process Signals

			Operand
Process element	**Design, Mode of operation**	**absolute**	*symbol*
Sensor	Keyswitch, NO	I 32.0	”Mainswit”
Sensor	Button, NC	I 32.1	”Emerstop”
Sensor	Button, NO	I 33.0	”Startwas”
Sensor	Pressure contact, NO	I 32.3	”In–pos”
Sensor	Limit switch, NO	I 32.4	”C–front”
Sensor	Limit switch, NO	I 32.5	”C–back”
Sensor	Limit switch, NO	I 32.6	”Doorop”
Sensor	Limit switch, NO	I 32.7	”Doorcl”
Actuator	Coupling relay	Q 32.0	”C–fwds”
Actuator	Coupling relay	Q 32.1	”C–bwds”
Actuator	Coupling relay	Q 32.2	”Open–d”
Actuator	Coupling relay	Q 32.3	”Close–d”
Actuator	Coupling relay	Q 32.6	”Rotate”
Actuator	Coupling relay	Q 32.7	”Shampoo”
Actuator	Coupling relay	Q 33.0	”Rinse”
Actuator	Coupling relay	Q 33.1	”Wax–on”
Actuator	Coupling relay	Q 33.2	”Dry”
Display	Lamp or display panel	Q 33.4	Car-in
Display	Lamp or display panel	Q 33.5	Car-out

Step 3:

Description of the process sequence, representation of the control functions in a decision table.

An important step in the program development is to establish the control sequence based on the schematic representations and the list of all the process variables. This can be achieved for example in the form of flowcharts.

A verbal description of the process sequence has been selected and the control task is solved using a decision table.

The decision table (Table KEIN MERKER) should be read as follows:

- The conditions that must be evaluated in a logical control step are listed above the double line and the actions that are executed if the conditions are fulfilled are listed below the double line.
- A column corresponds to a control number which is described verbally in the sequence and then programmed as a STL segment in step 5 of the program development.

Process sequence

1. Prepare for the program sequence.
2. Define the operating status.
   The control system defines the process status “on” when the main switch is on (I 32.0 = 1) and the PLC has started up (start–up ID in OB 20/21/22 = 1).
3. Switching off the process/stopping the carwash.
   To be able to stop the process at any time, e.g. in an emergency situation a safe switch off procedure is necessary: if the emergency stop button (I 32.1 = pulse) or the main switch is switched off (I 32.0 = 0) the control system resets the internal PLC status and deactivates all the outputs.
4. Moving the process to the initial position.
   When the control system starts up, the carwash is brought to its “initial position” if it is not already in this position. In the basic position, the door is open (I 32.6 = 1), the carriage with the brushes is at the back (I 32.5 = 1) and there is no car in the washing position (I 32.3 = 0).
   The control system must therefore check that these process statuses are correct. If not, the appropriate movements: “carriage backwards” (Q 32.1 = 1) and/or “open door” (Q 32.2 = 1) are started and if there is still a car in the carwash the display DRIVE CAR OUT (Q 32.5 = 1) must be lit up.

5. Establishing the conditions to start washing.
   The carwash status “initial position” is checked, i.e. the door is open (I 32.6 = 1), the brush carriage is at the back (I 32.5 = 1) and there is no car in position (I 32.3 = 0). This initial position is indicated by the display DRIVE CAR IN (Q 32.4 = 1). The display DRIVE CAR OUT (Q 32.5) goes off.
6. Driving the car in and starting the washing process.
   The car is driven into the washing position (I 32.3 = 1) and the driver leaves the car and goes to the control panel outside the carwash and presses the start button for the washing process (I 33.0 = pulse). After checking “car in position” (I 32.3 = 1) and “start button pressed” the control system closes the door (Q 32.3 = 1) and switches off the display DRIVE CAR IN (Q 32.4 = 0).

The following parts of the actual washing process including opening the door once the car has been cleaned run automatically without the driver taking any further action.

7. Applying shampoo.
   After the system checks the input signal “door closed” (I 32.7 = 1), the carriage moves forward (Q 32.0 = 1) with rotating brushes (Q 32.6 = 1) and the shampoo jets open (Q 32.7 = 1). The car is shampooed and brushed and the dirt loosened.
8. Washing, rinsing.
   After checking the front position “carriage front” (I 32.4 = 1), the control system switches off the frame drive (Q 32.0 = 0), closes the shampoo jets (Q 32.7 = 0), opens the water jets (Q 33.0 = 1) and moves the carriage backwards (Q 32.1 = 1) once again with the brushes rotating (Q 32.6 = 1). The car is cleaned and rinsed.
9. Applying wax.
   After checking “carriage back” (I 32.5 = 1) the drive is switched off (Q 32.1 = 0), the water jets closed (Q 33.0 = 0) and the brush drive switched off (Q 32.6 = 0).
   The carriage is now moved forward (Q 32.0 = 1) with the jets for applying wax open (Q 33.1 = 1).
10. Forming a wax film.
    When the front position is reached (I 32.4 = 1), the wax jets

are closed (Q 33.1 = 0) and the frame moved backwards again (Q 32.1 = 1).

11. Once the back position is reached (I 32.5 = 1), the drive is switched off (Q 32.1 = 0). The wax sprayed onto the car now requires a certain time (WT) to be distributed and to form a complete film on the surface of the car. The control system therefore waits until WT has elapsed. Once WT has elapsed, the next step of the process is enabled.
12. Drying the car.
    The drying process is initiated by starting the drying time DT and simultaneously opening the air valve (Q 33.2 = 1). When DT has elapsed, the air valve is closed (Q 33.2 = 0) and the door opened (Q 32.2 = 1).
13. Driving the car out.
    After opening the door (I 32.6 = 1), the door drive is switched off (Q 32.2 = 0) and the display DRIVE CAR OUT is lit (Q 32.5 = 1).
14. The carwash is empty.
    If there is no car in position (I 32.3 = 0) the control system switches off the display DRIVE CAR OUT (Q 32.5 = 0) and resets the internal step counter to zero.

The washing cycle is now completed. Once the car has been driven out, the carwash returns to the initial position (here, point 5) and the display DRIVE CAR IN is lit. The next car can be driven in and the washing process started again.

Note: The movement of the brushes to adapt to the height and profile of the car is not included in the example. This would be performed by a different subprogram.

The following diagram (Fig. KEIN MERKER) is a graphical representation of the process sequence. The numbers in brackets indicate the assignment to the process steps described and at the same time to the segment number in the decision table.

To separate one process step from another in terms of the program, an internal step counter is used. Once an operation is completed, the control system increments this counter by 1 and includes the current counter reading in the conditions for executing the next process step. The assignment and step counter reading are shown on the left in Fig. A-3.

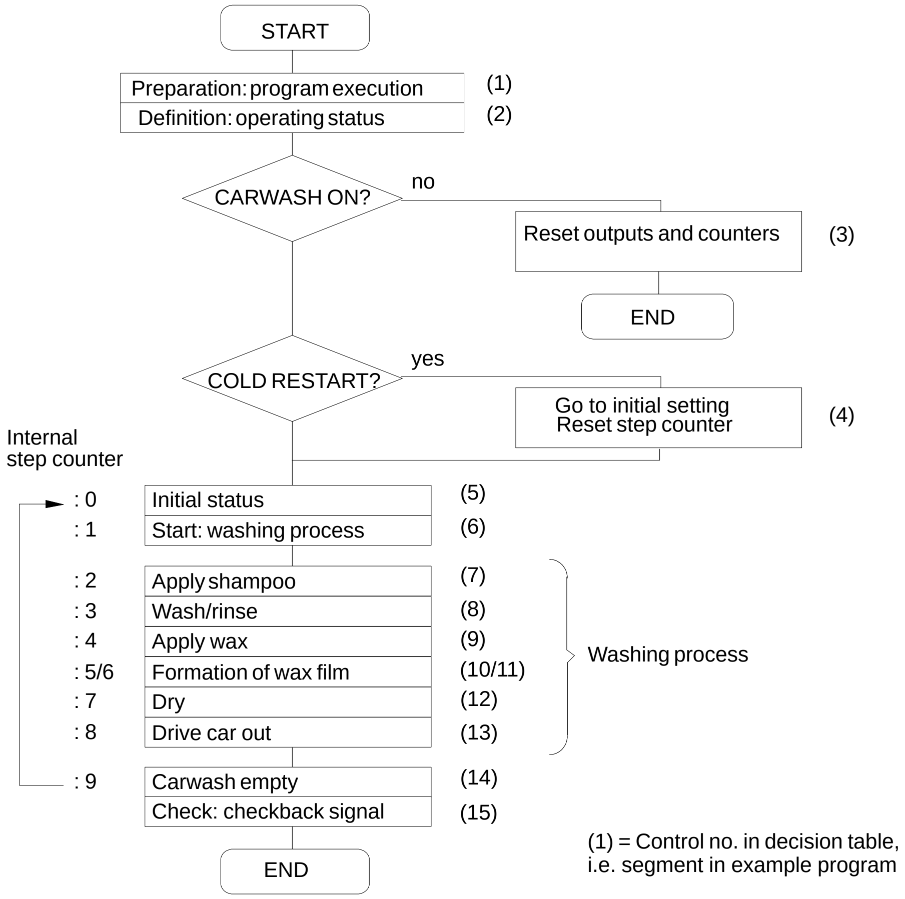


Figure A-3 Flowchart of a Carwash Process

Table A-2 Decision Table for the “Carwash” Program

OPERATIONS/actions	CONTROL no. (Segment)												
	2	3	4	5	6	7	8	9	10	11	12	13	14
Main switch/PLC start-up (OB20...22)	I 32.0	$\overline{\text{I 32.0}}$											
”Emeergency OFF” button		I 32.1											
”Start” (washing process) button					I 33.0								
Car in position			I 32.3	$\overline{\text{I 32.3}}$	I 32.3								$\overline{\text{I 32.3}}$
Carriage front (I 32.4), back (I 32.5)			I 32.5	I 32.5			I 32.4	I 32.5	I 32.4	I 32.5			
Door open (I 32.6), closed (I 32.7)			I 32.6	I 32.6		I 32.7						I 32.6	
Step counter for washing process				0	1	2	3	4	5	6	7	8	9
Pulse counter for switch on	→ M 10.1		M 10.1										
Counter reading KF Wax application time WT, drying time DT					KF + 1					→WT WT = 0	→DT DT = 0		
Display: DRIVE CAR IN				Q 32.4									
DRIVE CAR OUT			Q 32.5									Q 32.5	
Carriage fwd (Q 32.0), bwd (Q 32.1)			Q 32.1			Q 32.0	Q 32.1	Q 32.0	Q 32.1				
Open (Q 32.2), close (Q 32.3) door			Q 32.2		Q 32.3						Q 32.2		
Rotate brushes						Q 32.6	Q 32.6						
Apply shampoo						Q 32.7							
Wash/rinse							Q 33.0						
Apply wax								Q 33.1					
Dry											Q 33.2		
Carwash stop (reset outputs)													

Before we can move on to the next steps in creating the STEP 5 program, the program structure must first be established. Only a structured program can run on a PLC.

As simple as our example program may be, for it to run properly not only the program or function block with the control statements for the washing process and the corresponding data block are required, but also at least one organization block (OB 1). OB 1 is responsible for the cyclic execution of the program in the processor. In addition to this, the start–up blocks (OB 20/21/22) are also necessary. These are responsible for the cold or warm restart of the process under different conditions.

Without explaining the functions of the organization blocks in greater detail, Fig. KEIN MERKER illustrates the program structure with the block names as they are used in the example.

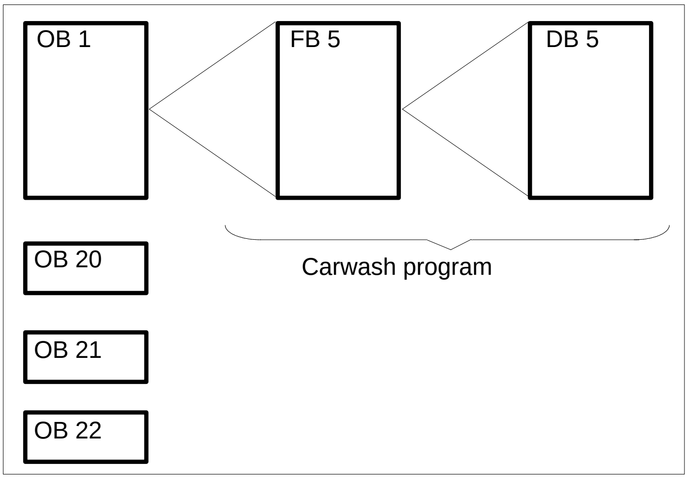


Figure A-4 Structure of the Carwash Program

Step 4:

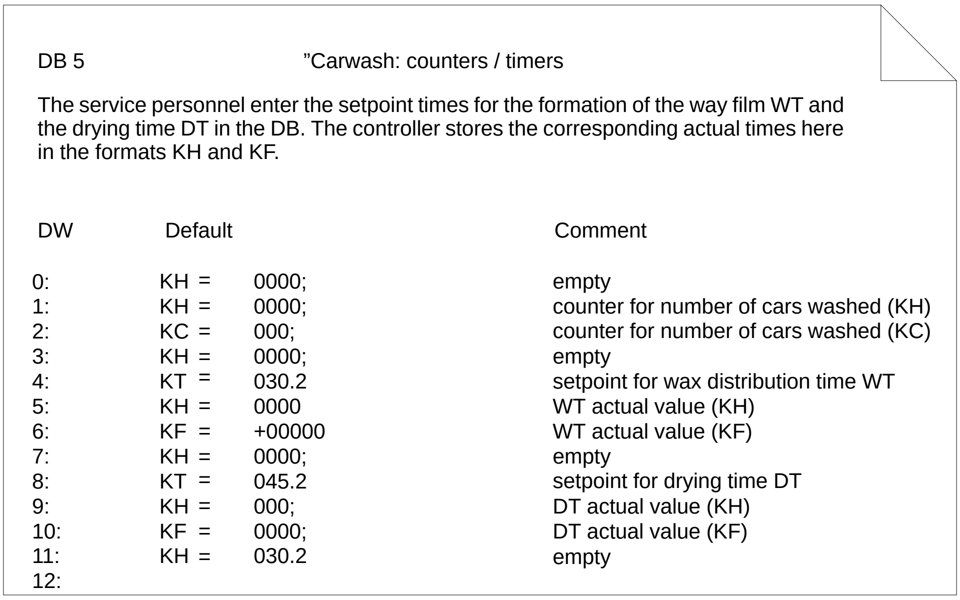
Specifying the data block

There are two further requirements for the control system:

– The service personnel should be able to change the times for wax distribution WT and the drying time DT.

– The number of washing cycles should be recorded and the number output when required.

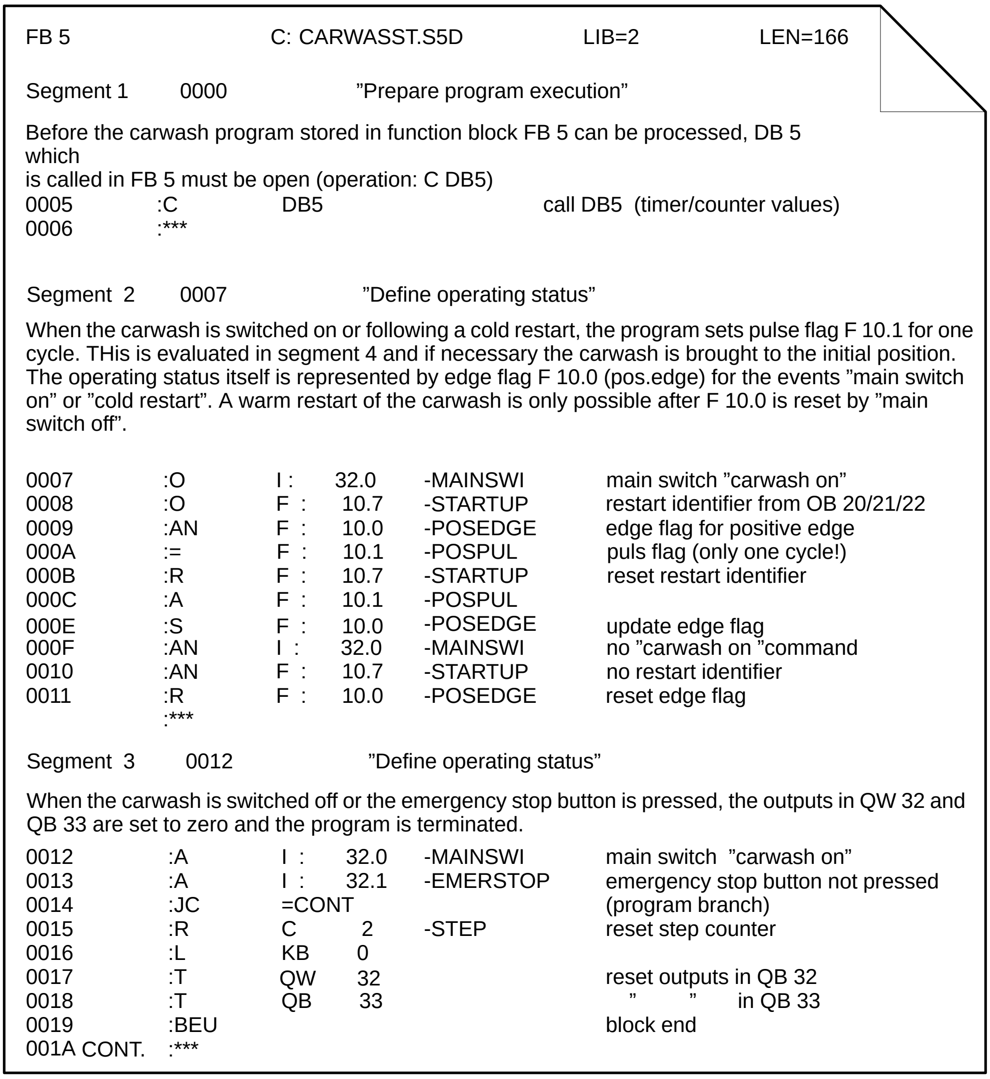
These functions are best implemented by setting up a data block (Fig. A-5). The data block contains the setpoints for WT and DT as well as the actual values of the timers in the formats KH and KF.



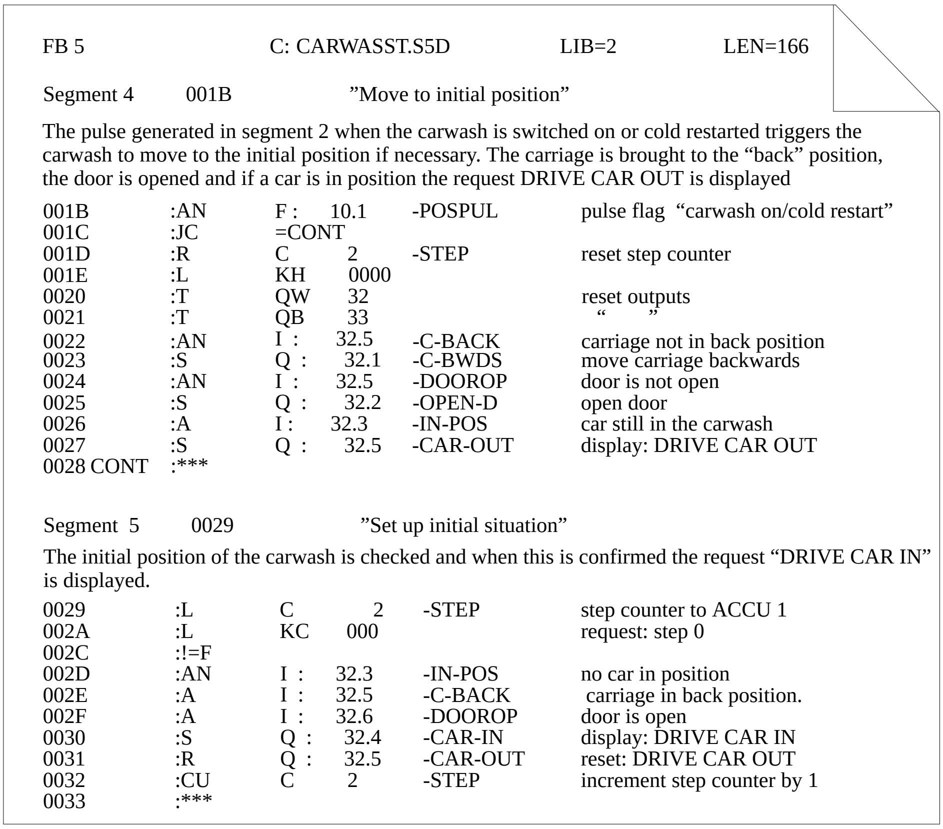
```
DB 5                        ”Carwash: counters / timers

The service personnel enter the setpoint times for the formation of the way film WT and
the drying time DT in the DB. The controller stores the corresponding actual times here
in the formats KH and KF.


DW          Default                          Comment

0:          KH =    0000;                    empty
1:          KH =    0000;                    counter for number of cars washed (KH)
2:          KC =    000;                     counter for number of cars washed (KC)
3:          KH =    0000;                    empty
4:          KT =    030.2                    setpoint for wax distribution time WT
5:          KH =    0000                     WT actual value (KH)
6:          KF =    +00000                   WT actual value (KF)
7:          KH =    0000;                    empty
8:          KT =    045.2                    setpoint for drying time DT
9:          KH =    000;                     DT actual value (KH)
10:         KF =    0000;                    DT actual value (KF)
11:         KH =    030.2                    empty
12:
```

Figure A-5 Contents of the Data Block for the Carwash (Printout)

Step 5: **Programming** (here only the first 5 segments)



```
FB 5                          C: CARWASST.S5D          LIB=2          LEN=166

Segment 1      0000               "Prepare program execution"

Before the carwash program stored in function block FB 5 can be processed, DB 5
which
is called in FB 5 must be open (operation: C DB5)
0005          :C          DB5                          call DB5  (timer/counter values)
0006          :***


Segment  2     0007                "Define operating status"

When the carwash is switched on or following a cold restart, the program sets pulse flag F 10.1 for one
cycle. THis is evaluated in segment 4 and if necessary the carwash is brought to the initial position.
The operating status itself is represented by edge flag F 10.0 (pos.edge) for the events "main switch
on" or "cold restart". A warm restart of the carwash is only possible after F 10.0 is reset by "main
switch off".

0007          :O          I :    32.0      -MAINSWI        main switch "carwash on"
0008          :O          F  :    10.7     -STARTUP        restart identifier from OB 20/21/22
0009          :AN         F  :    10.0     -POSEDGE        edge flag for positive edge
000A          :=          F  :    10.1     -POSPUL         puls flag (only one cycle!)
000B          :R          F  :    10.7     -STARTUP        reset restart identifier
000C          :A          F  :    10.1     -POSPUL
000E          :S          F  :    10.0     -POSEDGE        update edge flag
000F          :AN         I  :    32.0     -MAINSWI        no "carwash on "command
0010          :AN         F  :    10.7     -STARTUP        no restart identifier
0011          :R          F  :    10.0     -POSEDGE        reset edge flag
              :***

Segment  3     0012                "Define operating status"

When the carwash is switched off or the emergency stop button is pressed, the outputs in QW 32 and
QB 33 are set to zero and the program is terminated.
0012          :A          I  :    32.0     -MAINSWI        main switch  "carwash on"
0013          :A          I  :    32.1     -EMERSTOP       emergency stop button not pressed
0014          :JC         =CONT                            (program branch)
0015          :R          C        2       -STEP           reset step counter
0016          :L          KB      0
0017          :T          QW      32                       reset outputs in QB 32
0018          :T          QB       33                        "      "     in QB 33
0019          :BEU                                         block end
001A CONT.    :***
```



```
FB 5                    C: CARWASST.S5D          LIB=2          LEN=166

Segment 4     001B             ”Move to initial position”
```

The pulse generated in segment 2 when the carwash is switched on or cold restarted triggers the carwash to move to the initial position if necessary. The carriage is brought to the “back” position, the door is opened and if a car is in position the request DRIVE CAR OUT is displayed

```
001B          :AN       F :    10.1     -POSPUL        pulse flag  “carwash on/cold restart”
001C          :JC       =CONT
001D          :R        C        2      -STEP          reset step counter
001E          :L        KH       0000
0020          :T        QW      32                     reset outputs
0021          :T        QB      33                        “        ”
0022          :AN       I  :     32.5   -C-BACK        carriage not in back position
0023          :S        Q  :     32.1   -C-BWDS        move carriage backwards
0024          :AN       I  :     32.5   -DOOROP        door is not open
0025          :S        Q  :     32.2   -OPEN-D        open door
0026          :A        I :     32.3    -IN-POS        car still in the carwash
0027          :S        Q  :     32.5   -CAR-OUT       display: DRIVE CAR OUT
0028 CONT     :***

Segment  5    0029             ”Set up initial situation”
```

The initial position of the carwash is checked and when this is confirmed the request “DRIVE CAR IN” is displayed.

```
0029          :L        C          2    -STEP          step counter to ACCU 1
002A          :L        KC     000                     request: step 0
002C          :!=F
002D          :AN       I  :    32.3    -IN-POS        no car in position
002E          :A        I  :    32.5    -C-BACK         carriage in back position.
002F          :A        I  :    32.6    -DOOROP        door is open
0030          :S        Q  :     32.4   -CAR-IN        display: DRIVE CAR IN
0031          :R        Q  :     32.5   -CAR-OUT       reset: DRIVE CAR OUT
0032          :CU       C        2      -STEP          increment step counter by 1
0033          :***
```

The complete program including all comments and the assignment list can be found in the directory C:\S5_DATEN\DEFAULT under the name PROEXAST.S5D.

A.2 Glossary

A

absolute address
This is the physical address (number) of the memory location of an operand, at which it is accessed.

access rights, access protecion
With STEP 5, it is also possible to work from the PG via a bus link. The system manager then assigns attributes to the files: read only, read/write etc. These access rights to programs are set prior to editing in the ”*Settings*”.

actual operand
The actual operands (parameter list in the calling block) replace the formal operands in an FB/FX when it is called.

assignment list
List of assignments of absolute and symbolic operands and operand comments.
The assignment list is edited as a sequential file (*Z0.SEQ). When you save it, this sequential source file generates the → symbols file (*Zn.INI, n = 0,1,2).

B

block
A block is a section of a user program for a specific function, structure or use. In STEP 5, a distinction is made between blocks containing statements (OB, PB, SB, FB/FX) and blocks containing data (DB/DX) and variables blocks (VB) that are not used in the program but contain lists of variables for test purposes.

block body
The block body contains statements/logic operations in segments or it contains process data (in DBs).

block header
STEP 5 automatically sets up the header (length 5 DW) containing the start identifier, type and number of the block and the PG identifier, the library number and the block length (including the preheader).

block preheader In data and function blocks (DB/DX, FB/FX), STEP 5 generates an additional block header with the formats of the data used (DV/DX) or the identifiers of the jump labels (FV/FVX). The preheader is not transferred to the PLC or to EPROM/EEPROMs.

blow Transferring STEP 5 blocks to an EPROM/EEPROM submodule.

breakpoint To test sequences of statements in blocks, a breakpoint can be set. This is a point at which the RLO can be observed in the program (→ *Test, Block status/Status variable*). Program execution is stopped at the breakpoint and the signal states of the → actual operands are output.

buffer Temporary store to which selected program or text sections are written during editing so that they can be recalled and copied or transferred. The next buffer command overwrites the current content.

bus selection With the bus selection utility (*Management*, *bus paths*) connections from the PG to selected stations can be set up and activated. All STEP 5 functions can be performed via such a bus path just as with a point to point connection.

C

change STEP 5 menu for calling other S5 packages (e.g. GRAPH 5). It is possible to change to one of the loaded packages displayed in the COM selection box and then return to STEP 5 at any time.

comments

STEP 5 provides a wide range of possibilities for adding comments and explanations to programs. Comments are not transferred to the PLC. STEP 5 accepts statement, segment and plant comments. Since data blocks do not have segments, a block comment is created.

- Statement comments and line comments for DB/DXs (max. 32 characters) and segment titles (with DBs block titles) are stored in comment blocks (OC, PC, SC, FC).
- Segment comments and block comments for DB/DXs with a maximum of 16K characters are stored in documentation blocks (e.g. #PBDO.nnn). These are assigned to the ”program” block (PB, SB,FB etc.).
- The plant comment (explanation of the user program) is stored in an S5 documentation file with a freely selectable name (#DOCFILE, name = max. 8 characters).

compress memory

When blocks are deleted in the PLC, they are first declared invalid in the user memory. Whenever a block is corrected, an unaltered old block remains in memory. The STEP 5 function ”*Test, PLC control, Compress memory*” eliminates invalid blocks and closes the gaps between valid blocks to create more memory.

connector

An intermediate flag used to temporarily store the RLO (also inverted), so that the RLO can be used elsewhere avoiding repetitive logic operations.

control system flowchart, CSF

Representation of the logical relationships of a control task in the form of function symbols complying with DIN 40719, Part 6.

cross reference

If the function *”Management, Generate XRF”* is activated, STEP 5 generates the cross references to other uses of each operand and writes the references to a special program file *XR.INI. You can call up this information in the block editor (***F2** Reference*) covering more than one block.

cross reference list

This is created by STEP 5 from the set program file after the function has been selected in the ”*Documentation, Standard output or Enhanced output*”. The list contains the symbol for every absolute operand and indicates the blocks and segments where they occur.

cursor

The STEP 5 editors use a large cursor (known as the long cursor) and a small cursor. The long cursor indicates the current editing position in the editing field. It is displayed inversely and its length is generally the length of the actual input field. The small cursor is character oriented and is used for precise editing in the editing fields.

In LAD/CSF, the long cursor supports the graphical design of the segment in conjunction with the mouse. The cursor is moved within the grid of 8 columns and 50 lines (= 2.5 x screen height). In the ”small cursor” mode, no mouse operation is possible.

cursor control (automatic)

When using automatic cursor control, the cursor jumps to the next empty input field automatically when an entry is completed. The function switched on and off with the CURS key.

cycle time

The time required for the program to run through once in cyclic program execution. This time determines the maximum reaction time of a PLC to an external signal.

D

data block DB/DX

These blocks contain data (e.g. bit patterns, constant values) with which the program works. After it has been called, a data block remains ”open” until another data block is called.

directory

With the STEP 5 function *”Directory, in the program file”* or *”in the PLC or file”*, the block list of a program file is displayed or printed out. The block type, number, length and the library number of each block is displayed (not if PLC is selected).

documentation

The STEP 5 menu ”*Documentation*” provides functions for outputting program blocks and elements on a printer or to a file. In the ”Standard output”, the elements are output as they appear on the screen, in the ”Enhanced output”, graphical elements are added and a footer with user information is appended to each page.

documentation block

This contains the segment comments assigned to blocks (#OBDO.nnn, # PB.., #SB.., #FB..) and a block comment for data blocks (#DBDO.nnn).

E

editor

A software tool for creating → blocks in the form of Statement Lists (STL), Ladder Diagrams (LAD) or Control System Flowcharts (CSF) depending on the → settings. Special editors are used to create → data blocks, or → assignment lists and for writing segment and plant comments.

The STEP 5 ”*Editor*” menu provides access to the central tools for programming, creating blocks, designing logic controls and for acquiring process data.

EPROM/EEPROM

In terms of STEP 5, this is a utility which can be invoked under ”Management” and used to load (blow) and erase user programs in EPROM/EEPROM submodules.

F

flag

Flags are internal memory locations that can be addressed either bit or byte oriented (identifier F). Intermediate results of operations are written to flags.

footer

This is a labelling field appended to the bottom of each page printed out. The footer can be either 80 or 132 characters wide. This is selected in the *”Settings, Page 2”*.

formal operand

An operand that can be assigned parameters and that is connected to a substitution statement. In the FB/FX, only the operation to be performed on the operand is specified. The actual operand is substituted for the formal operand based on a parameter list when the block is called.

function block FB

This type of block contains programs or program sections (subprograms), particularly functions which are required frequently (→ standard function blocks) in the form of STEP 5 statements (basic and supplementary operations). FBs are intended for multiple use. The actual operands are transferred to the FB via the parameter list when it is called.

function element

A function element in LAD/CSF represents the relationship between ”input – processing – output” in a control system as a box with the signal flow ”conditions – function – operations”. STEP 5 recognizes binary function elements, e.g. ”&”, ”= >”, connectors, timers/counters and complex word–oriented function elements (digital functions) e.g. arithmetic, shift or convert operations. Owing to the different operand types, it is not normally possible to cascade binary and complex function elements.

function keys

These can have a fixed assignment (e.g. delete key, cancel etc.) or may be assigned functions appropriate to the current editor and situation (softkeys
F1 ...F8 – activated by pressing the keyboard key or clicking on the buttons at the lower edge of the screen).

I

input field

An operand field in LAD/CSF in which the operand with its type identifier and parameter or symbolic name (with hyphen) can be entered. An input field is ”undefined” when it contains 9 question marks. The field is ”not connected” when it can remain empty without an operand.

interrupt stack ISTACK

At each program execution level, the system program writes an entry in the interrupt stack whenever the PLC is interrupted, so that after the interrupt has been serviced, the program returns to the previous level. The information output (Test, PLC info) includes the address of the interrupt point with the current condition codes, the contents of the ACCUs and the cause of the interrupt.

I/Q/F list

This provides information about the bit assignments in bytes (W, DW) of the operand groups inputs (I), flags (F) and outputs (Q) *(Documentation, standard output, I/Q/F list and Enhanced output).*

job box

A dialog window for defining STEP 5 functions. Apart from naming the object to be processed, you can also select processing or output options.With the ”select” function in the job box a → selection box is displayed in which files and blocks etc. can be found and selected.

L

Ladder diagram (LAD) Graphical editing language for STEP 5 blocks in logic control programs, derived from circuit diagrams (DIN 19 239).

library number 5 digit number to identify blocks (block number)

long box → Function element

M

management The STEP 5 ”*Management*” menu provides functions with which the user program can be manipulated (generating cross references, rewiring operands, translating assignment lists etc.) and for storing blocks on EPROM/EEPROMs. This menu also includes an editor for creating path files for PG bus connections.

memory areas There are three memory areas in each PLC: the user area, the system area (BSTACK, ISTACK, address lists, counters, timers, flags, PII, PIQ) and the peripheral area (addresses of the process I/Os).

memory configuration STEP 5 function which diplays the amount of user memory occupied in the PLC.

N

new display When editing in LAD and CSF, this function (*half screen* key) reorganizes the screen and optimizes the display of the current segment, even when the operands are still incompletely labelled.

node Nodes are stations (PLC, PG, server) connected to a network. They are identified by a unique name. A bus path leads from the start node to (e.g. PG/AS511) via one or more nodes (e.g. CP) to an end node (e.g. CPU in the S5-135). Each node has a network address (node number).

O

object

An item which can be selected for processing in the STEP 5 ”*Object*” menu. According to this definition an object can be one of the following:

- a → project, i.e. the configuration of a user program
- a block, i.e. an editable and callable program module
- a PCP/M file that can be converted to an S5–DOS/ST/MT file or deleted.
- an S5-DOS/ST/MT file that can be converted to a PCP/M file or deleted.

operand

Process variable that can be addressed in absolute form (e.g. I 32.0) or in symbolic form (e.g. VALVE 1).

operand comment

These can be added to the symbols in the assignment list. They can be entered and modified directly in the block editor.

organization block OB

These contain STEP 5 operations (basic operations) particularly block calls. OBs are called by the operating system or by the user to call special functions and trigger certain reactions from the PLC. OBs are part of the user program and form the interface to the system program.

overall reset

Deletes all the blocks loaded in a PLC.

P

path file

A path file contains a selected (edited) bus path with all the node names and addresses. It is called using the required path name with the extension *AP.INI. The PG then establishes the path automatically.

PG link

Direct connection of two PGs via connecting cables.

plant comment

Text file for adding comments to a user program. This is not linked to a block. The file name must be preceded by the character #. The other 8 characters can be selected freely.

printer file

This file contains the parameters for the printer (formats, control sequences). It is named in the *Settings, page 2*. Its extension is *DR.INI and it is stored in the system directory.

process peripherals

All the sensors (limit switches etc.) required for process input and the actuators and indications required for process output.

process variable A process variable, also known simply as a variable, is an operand to which a process–dependent value can be assigned. These values can be variable or constant. The operands adopt a signal state.

program blocks → Blocks

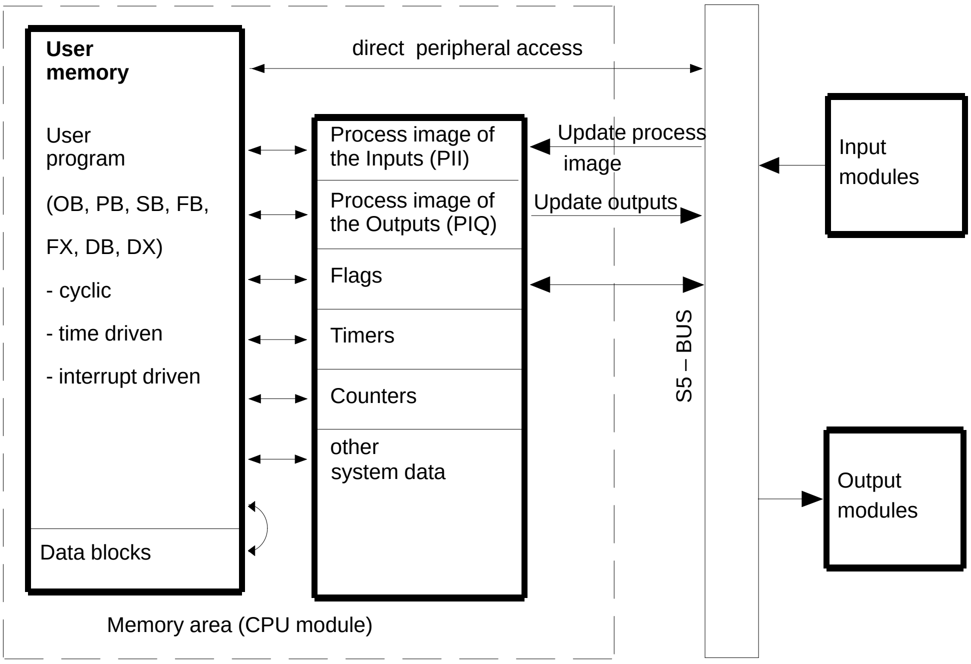
programming number This is used to identify the type of EPROM/EEPROM plugged in. This is assigned to the order number of the specific submodule. When a function is invoked (e.g. blow EPROM), STEP 5 examines the programming number and then displays the parameters of the submodule. This avoids errors when submodules are exchanged.

program structure Program overview display in which the nested calls of individual blocks is indicated starting from the OB (→ *Documentation, Standard output and Enhanced output*).

project The term ”project” (STEP 5 menu) is used to identify all the STEP 5 files belonging to one user program in a project file (*PJ.INI). This project file, which can be both loaded and saved, contains all the information, e.g. parameter settings and directory/file names for straightforward processing and maintenance of the user program.

process image If the operand groups I or Q are addressed by STEP 5 statements, the bits on the I/O modules are not scanned or modified directly, but rather a special area of the system memory in the PLC, known as the process image.

The process image of the inputs (PII) and outputs (PIQ) is processed and updated cyclically by the CPU. During start–up and at the start of every cycle, the signal states of the input modules are transferred to the PII. At the end of the program cycle, the CPU transfers the signal states in the PIQ to the output modules.


User memory
User program
(OB, PB, SB, FB,
FX, DB, DX)
- cyclic
- time driven
- interrupt driven
Data blocks
direct peripheral access
Process image of the Inputs (PII)
Process image of the Outputs (PIQ)
Flags
Timers
Counters
other system data
Update process image
Update outputs
S5 – BUS
Input modules
Output modules
Memory area (CPU module)

R

result of logic operation

The signal state at a particular point in the program, which is used for further binary signal processing. The RLO is the result of bit–oriented logic operations or the truth statement for comparator operations. It can, e.g. be combined with the status of operands or operations are executed depending on the previous RLO (e.g. conditional jumps). The RLO is in bit 1 of the condition code byte.

rewiring

This function assigns different or new addresses to operands in the user program. The function ”*Management, Rewire*” renames the operand in the whole program although the assignment only needs to be entered in a list once for each operand. Only the address and not the symbol is changed.

S

search

This function allows operands, segments or addresses to be located quickly within the program file. Before the function is started, the search key (identical the item to be found including upper and lower case letters) must be specified.

segment

A segment is a unit of a block which contains a sequence of logic operations (at least one) which implement a particular task and produce an intermediate result that can be used for further program execution. A segment can consist of any number of statements, however, in LAD and CSF, the number of operations is restricted to 6 or 7 owing to the size of the editing field on the screen. A segment is completed with ***.

segment identifier

To allow the editor to assign a segment comment to the correct segment, the editor automatically generates a 7–character string preceded by the $ character (e.g. $11___@). The number is the number of the segment. This identifier must not be modified or deleted otherwise the assignment of comment to block is lost.

selection box

A dialog window called in the job box for finding and selecting objects (blocks/files) in drives, directories and programs for processing with STEP 5.

settings

Settings box (2 pages) in the ”Object” menu to define a → Project in terms of naming the program files and selecting operating modes and type of representation on the PG/PC. All subsequent work in editing sessions relates to the selections made here.

SINEC H1

This is a bus system (network) for industrial environments complying with IEEE 802.3 (ETHERNET). PGs, PCs and PLCs can be connected. A bus segment has up to 100 stations connected and can be up to 500 m long. Segments are connected by repeaters. A maximum of two repeaters can be inserted between any two stations.

SINEC L1

This is a bus system for implementing small distributed automation systems with simple ressources. Only PLCs can be connected. A master PLC organizes the data traffic on the bus cable. The other PLCs are operated as slaves.

SINEC L2

This is a bus system based on the PROFIBUS standard (DIN 19245). There are both active and passive stations. Active stations can only access the bus when they have the token.The token is passed on in the logical ring in ascending order of the station addresses. Up to eight segments with a length between 0.2 and 1.2 km depending on the data rate can be connected via repeaters.

softkey

→ Function keys

standard function blocks

These are ready programmed ”off-the-shelf” function blocks for special applications. Each standard function block has a serial number assigned to it. The blocks represent self-contained functions that are required regularly in the user programs.

start address

The start addresses of all blocks in the user program are stored in the address list of DB2.

statement

The smallest independent unit of a program. It represents a task to be performed by the processor. A statement consists of the operation and the operand. The operand consists of the type identifier (e.g. I, Q, F, DW) and the parameter (e.g. 10.5, 25).

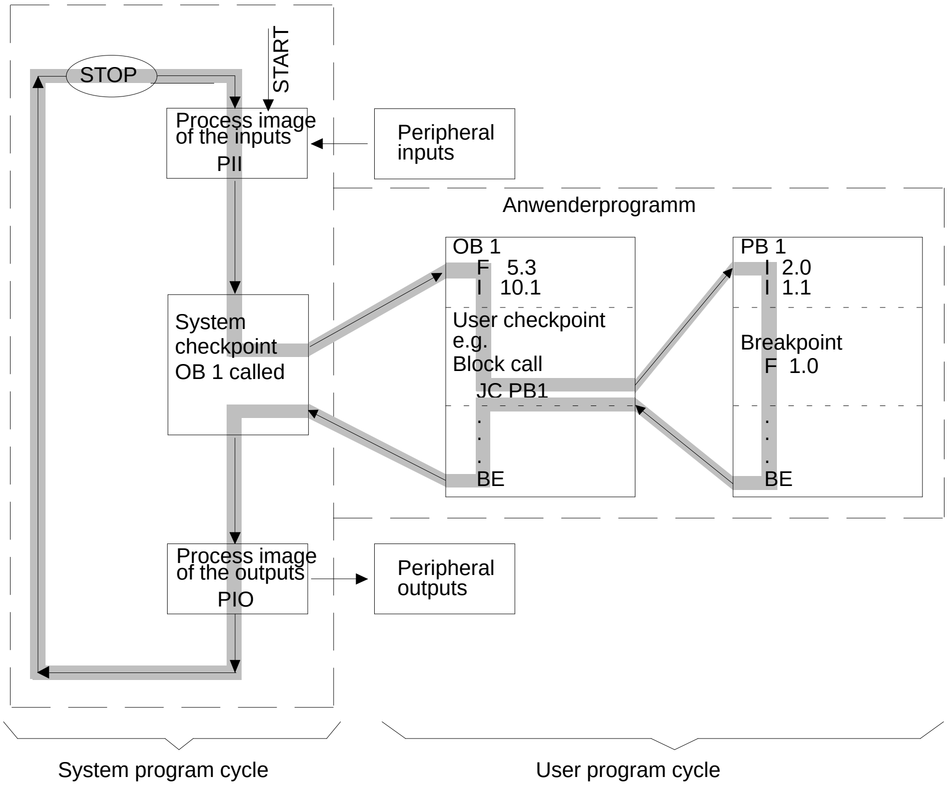
statement comment
This is an explanatory comment added to an STL statement. It is stored with the segment titles in comment blocks (OC, PC, SC, FC/FCX).

Statement List
An assembler-type alphanumeric input language for programmable controllers (DIN 19239) with one statement per program line. It can be used universally both for simple and complex control tasks. The statements are input and assigned addresses in the order in which they will be processed.

status
This function outputs the signal state of operands (bit 2 in the condition code byte). The status function is an online function and is selected in ”Test” menu.

symbols file
The list of the assignment of symbolic to absolute operands stored in a source file. Blocks programmed using symbolic operands are converted to absoulte address format with the help of the symbols file. They can then be understood by the processor.

system checkpoint
The system checkpoint is the interface between the operating system of the PLC and the user program. OB1 is called at the system checkpoint. In each cycle, the PLC operating system passes through the system checkpoint. At this point the process variables have the same state as the current process image. At the system checkpoint (Figure), the PG can be used to monitor or modify the signal states of the process variables and to set an output signal.



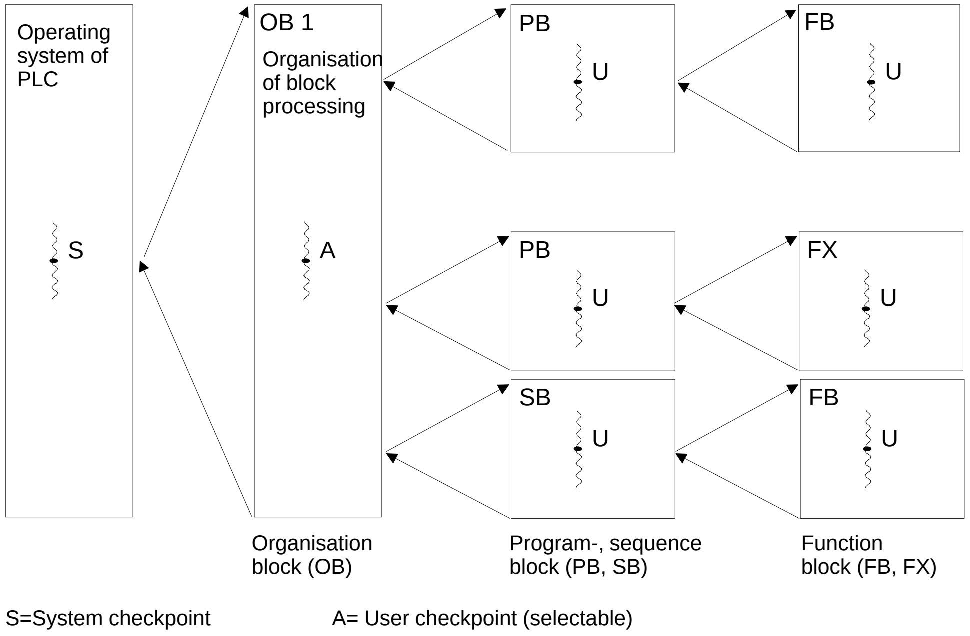
system identification file The SYSID file (*Settings, Page 1*) contains identification data and characteristics, e.g. for the communications processors (CPs).

SYSID system parameters These are data about the internal structure and the releases of the software and are located on every PLC. Information about the system paramenters is available with the function *”Test, Output PLC info”*.

test The STEP 5 menu ”Test” provides functions for testing user program blocks with the PLC online with the PG: these tests include logical and feasibility tests beyond the boundaries of one block. At the same time, information and correction functions are available depending on the PLC mode and the states of the process signals.

text editor Tool for creating and working with segments and operand comments in
→ Documentation blocks. Documentation blocks are called using the job/selection box in the STEP 5 block and the data block editor.

user checkpoint During program execution, process variables are changed dynamically and transferred to the process peripherals by the PLC at the end of each cycle. To be able to follow the changes to the variables while the program is running, the signal states of the variables can be output at any point in the program (Status variables or program test ON) (see Figure overleaf).



Checkpoints during program processing in the PLC.

V

variables block VB A variables block is used to store the content of the screen (operands, process variables) entered during the test functions status variables and the force functions → block.

wildcards * = substitute for a character string or a format-dependent name.
? = substitute for a character

XRF list → cross reference list

A.3 S5 Terminology

The following list provides you with the most common SIMATIC S5 abbreviations.

A

ABS	Absolute addressing, e. g. I 1.0
ACCU	Accumulator
ADF	Addressing error
AL	Assignment list
ARCNET	Attached Resource Computer NETwork. Network for the office environment
AS 511	Interface module 511, interface to the PLC

B

B	Block
BCD	Binary coded decimal number
BE	Block end
BSTACK	Block stack

C

C	Counter
COM n	Programmer software for a communications processor
COMP	Software program for comparing blocks
CP n	Communications processor (n = the number of the communications processor)
CPU	Central processing unit

CSF	Control system flowchart, graphical representation of the automation task using symbols complying to DIN 40 700/DIN 40 719

D

D	Data (1 bit),
DB	Data block
DBA	Data block start address
DBDO.nnn	Documentation block for a DB data block
DC	Comment block for a DB data block
DCX	Comment block for a DX data block
DD	Data double word (32 bits).When applied to diskettes: double density
DIR	Directory of the hard disk, diskette, PLC, EPROM and files
DL	Left data byte (8 bits)
DOCFILE	Documentation file, e. g. for plant comments
DR	Right data byte (8 bits)
DSP ABS	Presets screen form, display absolute operands
DSP SYM	Presets screen form, display symbolic operands
DV	Block preheader for a DB
DVS	Data management system for assignment lists
DVX	Block preheader for a DX
DW	Data word (16 bit)
DX	Extended data block
DXDO.nnn	Documentation block for a DX data block

E

EEPROM	Electrically erasable programmable read–only memory

EPROM	Erasable programmable read–only memory
ERAB	First scan (status bit)

F

F, FY, FW, FD	Flag bit, flag byte, flag word, flag double word
FB	Function block
FBDO.nnn	Documentation block for an FB function block
FC	Comment block for an FB function block
FCX	Comment block for an FX function block
FD	Default program file in which you are currently working. This abbreviation was used mainly in older STEP 5 software under PCP/M.
FlexOS	Operating system
FT	File transfer
FV	Block preheader for an FB
FVX	Block preheader for an FX
FX	Extended function block
FXDO.nnn	Documentation block for an FX function block

G

GRAPH 5	Software package for planning and programming sequence controls in a clear graphic representation (optional package)

I

I, IB, IW, ID	Input, byte, word, double word
IP	Intelligent peripheral module
ISTACK	Interrupt stack

K

KOR	Coordinator module
L	
LAD	Ladder diagram, graphical representation of the automation task with the symbols of a circuit diagram complying with DIN 19239
LEN	Length of a block
LIB	Library number
N	
NAU	Power failure
O	
OB	Organization block
OBDO.nnn	Documentation block for an organization block
OC	Comment block for an organization block
OY, OW	Byte, word from the extended peripherals
P	
PB	Program block
PBDO.nnn	Documentation block for a program block
PC	Comment block for a program block
PCP/M-86	Personal CP/M–86 operating system
PG	Programmer
PG-NET	Software package for connecting programmers to a network (option)
PI	Process image
PII	Process input image
PIQ	Process output image

PLC	Programmable controller
PW	Peripheral word
PY	Peripheral byte
Q	
Q, QB, QW, QD	Output, byte, word, double word
QVZ	Timeout
R	
RAM	Random access memory
REW	Rewire, renaming inputs and outputs in the user program
RLO	Result of logic operation (bit condition code)
RO	Read-only access
RW	Read and write access
S	
S5-DOS/MT	S5 operating system
S, SY, SW, SD	S flag, extended flag, byte, word, double word
SAC	Step address counter
SB	Sequence block
SBDO.nnn	Documentation block for a sequence block
SC	Comment block for a sequence block
SINEC H1	Bus system, network for an industrial environment
SINEC L2	Bus system based on PROFIBUS standards
STA	Status (bit condition code)
STEP 5	Programming language for programming SIMATIC S5 programmable controllers

STL	Statement list, STEP 5 method of representation using mnemonics of the S5 operations (complies to DIN 19239)
SYM	Symbolic addressing (e.g. –INPUT)
SYSID	Block for system identification
T	
T	Timer
V	
VB	Variables block
X	
XRF	Cross reference list (file *XR.INI)
Z	
ZYK	Cycle error

A.4 Key Assignment

The keyboard of a personal computer can have different functions assigned to the keys, i.e. the key functions depend on the currently active software. This also applies to the STEP 5 software. As soon as you load STEP 5, the keys take on specific S5 functions. There are two types of keys:

- dynamically assigned keys (function keys)
- keys with a fixed assignment

Dynamically Assigned Keys (function keys)

The keys F1 to F8 are known as function keys. Depending on the software level at which you are currently working, these keys are assigned the functions that are possible and also required at this level. The function keys are displayed in the menu at the lower edge of the screen. Some of the keys have a double assignment, function keys ***F1*** to ***F8*** and ***SHIFT F1*** to ***SHIFT F8***.

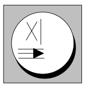
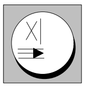
Keys with a Fixed Assignment

Such keys always have the same function, e.g. within STEP 5, the HELP function or the cursor control. These can also have multiple uses in combination with the SHIFT, ALT or CTRL keys.

Keys in LAD/CSF

Function Control Keys

Key name	Key	Output	Edit	Remarks
HELP	HELP	Displays a help text on the screen	Displays help information	Also available with ***SHIFT F8***
Hardcopy	PRINT	Prints out the whole screen on a printer or to a file	Prints out the whole screen on a printer or to a file	

Key name	Key	Output	Edit	Remarks
Half screen	PAUSE	Disabled	New, optimised screen display	In ”edit” also with ”extras” ***(SHIFT F7)*** and ***F2*** ”New disp”
Zoom-in	CTRL END	Disabled	Changes to ”symbol correction”	In output, only available with ***F1***. In ”edit” also with ”extras” (***SHIFT F7***) and ***F2*** ”New disp”
Editing mode	5 CORR	Changes to the editing mode (correction)	Disabled	In ”output” also with ***F6***.
Segment comment	÷ COM	Changes to the comment input mode – branch to segment title or segment comment	As output	In ”output” and ”edit” also with ***SHIFT F6***.
Insert segment		A segment is inserted before the current segment. An empty screen is displayed and you change to the editing mode	Disabled	In ”output” also in ”segment functions” with ***F5*** (Insert)
Delete segment	SHIFT 	Deletes the displayed segment. The segment is not buffered.	Disabled	In ”output” also in ”segment functions” with ***SHIFT F4***. In ”segment functions” the segment is written to the buffer file.

Terminating Keys

Key name	Key	Output	Edit	Remarks
Cancel (escape)	ESC	Changes back to the previous level	Modifications within a field can be cancelled. Otherwise you change to ”output”. Newly entered segments are deleted.	If you exit ”edit” the segment is displayed in its old form. If the segment has been input as a new segment, the previous one is displayed. Also with ***F8***.
Insert	0 Insert	Stores the currently displayed block if it has been changed. Changes back to the calling level.	Stores the currently edited segment. Displays the segment in its newest form.	Same as ***F7***.
Return	↵	Disabled	Completes input in fields. In empty fields the cursor is moved one field to the right.	
Enter segment	ENTER ***	A new segment is inserted after the segment displayed. An empty screen is displayed and you change to the editing mode.	Enters the segment you are currently working with and opens a new segment.	In ”edit” also with ***F6***.

Control keys

Key name	Key	Output	Edit	Remarks
Page up	9 ↑↑↑	Moves the displayed segment one line up.	As ”output”	In selection boxes one page up.
Page down	3 ↓↓↓	Moves the displayed segment one line down.	As ”output”	In selection boxes one page down.
SHIFT Page up	SHIFT 9 ↑↑↑	Moves the displayed segment one page down.	As ”output”	
SHIFT Page down	SHIFT 3 ↓↓↓	Rolls the displayed segment one page up.	As ”output”	
Page one segment forwards	+	The next segment is displayed.	Jump to the end of the current line.	In ”output” also in the ”segment functions” with ***F2***.
Page one segment back	−	The previous segment is displayed.	Jump to the start of the current line	
Segment end	SHIFT +	Disabled	Jump to the end of the displayed segment.	
Segment start	SHIFT −	Disabled	Jump to the start of the displayed segment.	
Input field end	⇄ − TAB +	Disabled	Jump to the end of the input field on which the cursor is positioned.	

Key name	Key	Output	Edit	Remarks
Input field start	SHIFT ⇄ TAB	Disabled	Jump to the start of the input field on which the cursor is positioned.	
Horizontal expand	7	Disabled	Expand the segment by one column at the cursor position.	Not permitted at the left margin of a LAD segment. In ”edit” also with ***SHIFT F7*** = Extras as ***F6*** = ***Hor exp***
Vertical expand	1	Disabled	Expand the segment by one line at the cursor position.	Not permitted in the two top lines of LAD segments.
Delete character marked by cursor	DEL	Disabled	Deletes a single character marked by the cursor.	
Delete subfield	SHIFT DEL	Disabled	Deletes a whole subfield.	
Delete character left of cursor	⇤	Disabled	Deletes a single character to the left of the cursor.	
Cursor right	⇒ 6 →	Positions the cursor on the next input field to the right. At the end of the line jumps to the first position in the line.	As ”output”. Within the input field you can also move the cursor to the position right of the short cursor.	
Cursor left	⇐ 4 ←	Positions the cursor on the next input field to the left. At the start of the line jumps to the last position in the line.	As ”output”. Within the input field you can also move the cursor to the position left of the short cursor.	

Key name	Key	Output	Edit	Remarks
Cursor up	8 ↑	Positions the cursor on the input field above the long cursor.	As ”output”	
Cursor down	2 ↓	Positions the cursor on the input field below the long cursor	As ”output”	
Change to input field	SHIFT ⇒ 6 →	As cursor right	The editing mode to modify the input field is activated. Empty input fields are deleted with this mode change. This key completes the input field and moves on to the next field to the right.	
Change to next input field	SHIFT ⇐ 4 ←	As cursor left	Completes the input field, moves to the next input field to the left.	

Special Keys

Key name	Key	Output	Edit	Remarks
Connector = ***F9***	F9	Disabled	Inputs a connector at the current cursor position	Also ***F5*** = ***Binary op*** and ***F4*** = ***#***
Negated connector = ***F9***	SHIFT F9	Disabled	Inputs a negated connector at the cursor position	Also ***F5*** = ***Binary op*** and ***F5*** = /.
Not defined ”?”	?	Disabled	Input fields are marked as undefined when this key is pressed first after selecting the input field	

Key Assignment STL

The following tables only explain the key assignment when the functions are different from those for LAD or CSF. All other key functions are listed under → *Key assignment LAD/CSF.*

Key name	Key	Output	Edit	Remarks
Cancel (escape)	ESC	Return to previous level.	Delete newly input segments	
Half screen	PAUSE	Changes the comment mode between operand and statement comments.	As ”output”	Also ***SHIFT F4***
Segment comment	COM ÷	Changes to the input mode for segment title, if pressed twice to the segment/block comments.	As ”output”	In ”output” also with ***SHIFT F6***.

Control Keys

Key name	Key	Output	Edit	Remarks
Cursor right	⇨ 6 →	Disabled	Positions the cursor to the right within an input field. At the end of the field jumps to the first position of the next input field.	
Cursor left	⇦ 4 ←	Disabled	Positions the cursor to the left within an input field. At the end of the field jumps to the first position of the next input field.	
Change to next input field	SHIFT ⇨ 6 →	Disabled	As output	
Change to next input field	SHIFT ⇦ 4 ←	Disabled	As output	

A.5 Brief Operating Instructions

Job Boxes

The majority of selectable functions must first have parameters assigned and then be activated. You assign parameters after calling the function in job and selection boxes.

Within these boxes, you can move the cursor with the ***mouse*** or the ***tab*** key and the ***cursor*** keys. In certain fields (colored or inverse display) you can call further selection boxes with ***F3*** = *Select*.

Object

This menu provides functions with which you can organize your program and files.

Project

You make all the required settings for a program once and store them in a project file (*PJ.INI). The following must be set, e.g.:

- storage location for the various files
- files involved
- method of representation (LAD/CSF/STL)
- mode
- parameters for printing out
  etc.

Settings
page 1
page 2

In the displayed “settings boxes” you enter the files and parameters. This box is divided into two pages. The selected parameters are set in the corresponding job and selection boxes. The set files and parameters on these two pages are valid for all the work in the project. In the “settings box” you can position the ***cursor*** using the cursor keys or the mouse. By double clicking on the parameters, you can either open a selection box or change the default. You can also achieve this by pressing the ***F3*** key twice.

Save	This saves all the settings made in the “settings” boxes in the current project file (***PJ.INI**).
Save as	Save the settings in a new (selectable) project file (***PJ.INI**).
Load	A file created as described above is loaded. All the settings contained in the file are then valid. Existing settings are overwritten.
Blocks	Here, you manage blocks and documentation files on the PG or the PLC. The following functions are available:
Directory	This outputs a directory on the output device selected in the job box (PG-PLC).
Transfer	Transfers blocks and documentation files from file to file, file to PLC, PLC to file. You select the source and destination in the displayed job boxes.
Compare	You can compare single blocks with each other, single blocks of a group of blocks or all blocks of a program file with a second program file. You can compare file with file, file with PLC, PLC with file.
Delete	This is used to delete blocks on the PG and PLC and documentation files only on the PG, PLC overall reset.
DOS files	With this function you manage files without having to change to the operating system level. You select a directory or search for a particular file in a directory using the job box. The following functions are available:
Directory	This lists the contents of a directory.
Copy	You can copy single files or groups of files.
Delete	You can delete single files or groups of files.

PCP/M files	With this function you can handle PCP/M files.
Directory	A directory created under PCP/M is displayed in the “directory of PCP/M files” job box, depending on your specifications.
Copy	This converts PCP/M files to S5–DOS ST/MT files. It can also be used to convert STEP 5 files created with S5–DOS ST/MT into PCP/M files.
Delete	PCP/M files on a PCP/M medium are deleted.

Editor

Using this menu you can start various program editors.

STEP 5 block	With this function, you can start the LAD/CSF or STL editors. The job box “edit STEP 5 block(s)” is displayed. Here, you select a block. The editor selected in the “settings” is then displayed.
Data blocks	With this function you supply parameters and start the editor for data blocks.
DB screen forms	With this function you supply parameters and start the editor for DB screen forms.
Assignment list	As soon as you activate this function, the editor for the sequential source file is called directly.

Test

With this menu, you activate the test, information and start–up functions with the PG in the online mode. There must be a physical and logical connection between the PG and PLC. You create this connection in the “settings” boxes using the “mode” field.

Block status	With this function you can test and correct blocks loaded on the PLC. You select the block to be tested in the “block status” selection box.
Status variables	With this function you output the current signal states of selected operands at the system checkpoint during program processing. You edit the operand list in an empty table.

PLC control With this function you can start and stop a PLC connected online or compress the user memory in the PLC.

Start PLC With this function you trigger a cold or warm restart on the programmable controller.

Stop PLC This changes the PLC to the STOP mode.

Compress memory With this function you can eliminate invalid blocks on the PLC and shift the valid blocks together.

Force variables With this function you can modify process variables and intervene in the process. You edit an operand list in the displayed table.

Force outputs With this function you can set outputs to on or off. The PLC must be in the STOP mode.

Output PLC info You can obtain information about the status of the connected PLC.

ISTACK A table of the control bits and their current status is displayed on the screen. With the PLC in the STOP mode, the interrupt stack is output to allow you to analyze the cause of an error.

BSTACK This provides you with information about the start address of the currently valid block and the relative and absolute return address in the block stack.

Output memory contents This function outputs the absolute addresses of the PLC and their contents on a selectable medium.

Memory configuration This outputs the memory configuration and indicates how much of the user memory in the PLC is currently occupied.

System parameters This displays the system parameters of the PLC on the screen.

Program test ON With this function, a block in the PLC is processed step by step. You select the box in the “program test ON” selection box which you can then manipulate or search for an operand that you want to monitor.

Program test OFF This switches off the program test function.

Management

This menu provides you with a series of utilities required in many situations when working with the STEP 5 editing and test functions.
The presets for the individual functions must already be made in the “settings” boxes.

Generate XRF — This generates a cross reference list for the set program file. As soon as you activate this function, a cross reference list is generated.

EPROMs — With this function you can transfer (blow) STEP 5 programs from the selected program file to EPROM/EEPROM submodules. The “presets” box is displayed.

Rewire — With this function you can rename operands and assign them to different outputs.

Automatic — The operands are renamed automatically based on a modified or new assignment list. The job box “automatic rewiring” is displayed. Here, you select the new program file name “to program file” and “with new symbols file”. The function is then executed immediately.

Manual — You rename operands in an operand list. The job box “manual rewiring” is displayed. Here, you select the new program file name “to program file”. Following this, you enter the operands in a table.

Assignment list — With this function you can process the assignment lists required for symbolic addressing of operands in your user program.

Convert SEQ >INI — You convert the sequential source file to the corresponding symbols file. You enter the name of the source file to be converted in the “convert SEQ > INI” job box.

Convert INI >SEQ — You convert the symbols file to the corresponding sequential source file, and you can have this sorted according to absolute or symbolic operands. You enter the name of the symbols file to be translated and the type of sorting you require in the “convert INI > SEQ” job box.

Correct INI — With this function you can change the name of the symbols file to be corrected. You enter the name of the symbols file to be corrected in the “correct INI” job box. Following this, you can correct the symbols file.

Convert stage V1.x V2.x	Symbols files created with earlier versions (V1.0, V2.0) can be converted.
Delete SEQ	This function deletes a sequential source file.
Delete INI	You delete the symbols files (*Z0.INI, *Z1.INI, *Z2.INI).
Output error list	You output the error list created if errors occurred during the conversions.
Select drive	With this function you can select the drives on which, for example, further S5 packages are installed.
Bus paths	With this function you can create, store and activate connections that are not established as point–to–point connections. You can create bus paths in the “select function/presets” selection box.

Documentation

This menu provides a selection of functions with which you can output programmable components such as blocks, files and lists on printers or to files. You can also evaluate certain data according to various criteria.
You can select either the standard output, i.e. output as edited or the enhanced output function. Enhanced output corresponds to the functions previously provided by KOMDOK.
The footer file and printer parameter file must be defined in the “settings” box.

In the “documentation” menu, you also select the printer parameters and the footer.
How to move the cursor in the job box to assign parameters for the output function is described briefly in this appendix in → *Job box*.

Standard output	With this function, you output program components in their basic form (as you edited them) either on a printer (A3, A4), to files or on the screen. At the same time, you decide whether the information is output from a program file or from the PLC.
Program structure	This outputs the call identifiers of the individual blocks of a program file. You select the required blocks in the “output program overview” job box.

STEP 5 blocks	You output the blocks of a program file in the methods of representation LAD, CSF and STL with or without cross references and with or without diagnostic SP data. You select the output you require in the “print STEP 5 blocks” job box.
Data blocks	You can output either individual or all the data blocks of a program.
DB screen forms	This function outputs data blocks containing screen forms. Select the blocks in the “DB SCREEN FORMS print blocks” selection box.
Assignment list	You output an assignment list. If the assignment list is not already set, you can select it in the “SYMBOLS: print SEQ file” selection box.
XRF list	You generate a cross reference list from an existing program file. Select the required operands in the “output XRF list” job box. A cross reference file does not need to exist.
I/Q/F list	You output an I/Q/F list. Select the required group of operands in the “output I/Q/F list” job box.
Three-in-one	With this function you output the program overview, the I/Q/F list and cross reference list with one command.
Enhanced output	This function, previously known as KOMDOK, allows you to document STEP 5 programs comprehensively and with little effort using DOC commands. In contrast to the standard output, the printouts have graphics added to them. Using DOC commands, you can structure the printout for your needs.
Program sections	In this menu, you can activate the output of the following files:
Blocks	You print out blocks of a program file in the methods of representation LAD, CSF and STL with or without cross references and with or without diagnostic SP data.
Block list	This function prints out a list of all the program and data blocks of the set program file.
Assignment lists	You output an assignment list. You can print this either in sequential form as edited or sorted according to absolute/symbolic operands.
Reference data	In this menu, you can activate the print out of the following lists or data:

Program structure	This prints out the call structure of the individual blocks in a program file.
XRF lists	This prints out cross references from a cross reference list according to specific criteria.
I/Q/F list	This prints out an I/Q/F list. The I/Q/F list shows you which bits in which bytes of the operand groups F, I, Q are assigned.
Checklist	This function checks through the programming data. Depending on the operation, the free operands, operands without symbols, and operands without setpoint data for the I/Q/F operands.
Text files	You can print out *LS.INI files or any ASCII files.
DOC commands	You can control all printouts made with the enhanced output function using DOC commands. These commands are put together like a program, stored in a file and started when the file is called. You can also call up other DOC command files using an appropriate statement in a DOC command string. This allows you to structure the printout.
Edit	You edit DOC commands and store them in a file.
Check	The DOC commands are checked to ensure that they can be run. If errors are detected, the exact cause of the error is written to an error file. If there are no error messages, no error file is created.
Output error list	Errors detected executing the “check DOC command” or “execute DOC command” functions are output.
Execute	The DOC commands in a file are started.
Print	This prints out a DOC command file.
Edit structure	This provides you with information about the connections between individual DOC command files. At the same time, you can edit individual DOC command files.
Print structure	The structure of interconnected DOC command files is output graphically.
Settings	Before you can print out the various files, you must select the printer parameters and the footer.

Printer parameters	You create a control character record for your particular printer which is stored in a printer file.
Edit footer	With this function you can create a new footer file or modify an existing file.

Change

In this menu, you can change to other S5 packages. These packages must be installed in a directory on one of the drives. You can then change to one of the S5 packages displayed. Once you select another package, you exit the STEP 5 user interface. You can change back to the STEP 5 interface, however, from every other S5 package.

further

You select the S5 package you want to activate in the “other SIMATIC S5 programs” selection box.

Help

With these functions, you can display the following information:

Key assignment list

This displays information about the “acceleration keys”. These are function keys with which you can activate certain functions directly.

Info-STEP 5 version

This provides information about the current STEP 5 version you are using.

Version of S5 packages

A list of the individual program components of the STEP 5 software is displayed.

A.6 PG Link between two PGs

The task of the PG link package is the exchange of STEP 5 blocks or files between various programmers.

The compatibility is ensured despite different diskette formats (40 or 80 tracks) diskette sizes (3 1/2” or 5 1/4”) and recording densities.

Hardware Requirements

Data exchange with the partner PG is only possible on an **active** TTY interface (20 mA).
If the existing COM1 port is only equipped with a V.24 or passive TTY interface, the S5 interface must be emulated. To do this, a converter (Köster box) is connected between the PG and the connecting cable to the partner PG. This converter converts the V.24 interface of the PG to an active TTY interface and therefore simulates the S5 interface of an S5 programmer.
You connect your PG with the partner in one of the two following ways:

– Via the **active TTY interface COM 1**
The PG and the partner PG are connected via two connecting cables.
or

– Via the **passive TTY or V.24 interface COM 1**
If you have a PG with a passive TTY interface or with only one V.24 interface COM 1 the passive interface must be converted to an S5 interface using a Köster box.
The PG is connected to the Köster box via a connecting cable.
The Köster box is connected to the partner PG via a further connecting cable.

The connecting cables are described in the PG 7xx manuals.

Loading the PG Link Package

Load this package using the STEP 5 function ”change” (→ *Change*).

As soon as you have activated the PG link package, this is started and you immediately switch to the PG link user interface.

PG Link

With the PG link package you can perform the following functions:

- Switch the PG passive. For data exchange an active and passive PG are required.
- Send data from the active to the passive PG.
- Fetch data from the passive to the active PG.

Entering the Presets

Once you have activated the PG link, the ”PRESETS” box is displayed. Here, you select the ”program file” (all the block specifications you make refer to blocks in this program file). You move to this file with ***SHIFT*** and the *cursor* keys.

The fields ”path file” and ”path name” are not relevant. Within the box you can make the following entries:

Input field	Explanation
F3 = *Select*	The cursor only jumps to the position at which you can make an input after you press the ***F3*** key.
F6 = *Enter*	The parameters you input are entered and you call ”function selection”. The ***Enter*** key has the same effect.
F7 = *Info*	You obtain information about the field marked by the cursor.
ESC = *Cancel*	Return to STEP 5 without any action being taken.

Function Selection

As soon as the presets have been entered (***F6***), the ”SELECT FUNCTION” box is displayed. You can make the following inputs:

Key level 1	 2	Effect of the function keys
F1		PASSIVE This switches the programmer from the ACTIVE to PASSIVE status. The PG to which data are sent must always be PASSIVE. The passive setting is cancelled by pressing ***ESC***.
F3		SEND You switch to the next key level in which the data exchange is activated.
	F1	BLOCK (send) The command line: **BLOCK:** **SEND TO PARTNER** appears. You can make the following inputs in the ”block” field.
		e.g. Explanation PBx Single blocks #DOC Documentation files FB Blocks of one block type * Various blocks from a block list A All blocks of the preset program file # All DOC files empty All blocks and DOC files Complete your input with the ***Enter*** key and the transfer to the partner PG begins automatically.
	F2	FILE (send) The command line **FILE:** **SEND TO PARTNER** **DEST DR:** appears. Here, you enter the file names to be transferred: X:NNNNNNNN.EEE (maximum 8 characters before the period). e.g. C:PROGFILE.S5D DEST DR: here, you enter the required drive. Complete your input with the ***Enter*** key and the transfer to the partner PG begins automatically.

Key level 1	 2	Effect of the function keys
	F5	P-DIR This outputs the directory of the partner PG. The command line **OUTPUT DIR FROM PARTNER BLOCK:** appears. Here, you enter the blocks as described under ***F1***. A block list (*) cannot be selected. Complete your inputs with the ***Enter*** key and the display of a block list is started automatically.
	F6	P-PRG.DAT With this you can set the program file of the partner PG. The command line **SET PRG.FILE PARTNER FILE NAME: ST.S5D** appears. Type in the required file name. When you complete your inputs with the ***Enter*** key, the file is set.
F4		FETCH This is effectively the same function as ”SEND”, however, you transfer the files or blocks from the passive to the active PG.
F6		PRESETS The ”presets” box is displayed
F7		AUX FCT With this function you can manage blocks and documentation files and select program files. You can perform the following functions: • Transfer blocks and documentation files (***F1*** TRANSFER) • Delete blocks and documentation files, overall reset of the PLC (***F2*** DELETE) • Output a directory (***F3*** DIR) • Change the preset program file (***F6*** PRG.DAT)
F8		RETURN Return to STEP 5

A.7 Key Macro

Using the key macro program, you can record key sequences in the **block editor**. The sequences are saved in the S5 file *TP.INI. This file is on the drive in which STEP 5 was installed. You will find the file in the directory selected in Settings page 1 for this drive.

Entering a key macro	Key macro permitted offline 1. Press key combination ***CTRL + A*** 2. **RECORDING KEY SEQUENCE** **MACRO NAME:** `@@@@@@TP.INI` 3. Name of the key macro: max. 6 characters. During recording the mouse must not be used. **Note** **Continuing after a message** Once the PG has output a message relating to the key macro, you can only continue working after entering a file name.

Completing entry of the key macro and saving	Press key combination ***CTRL + E*** **Note** **Make a note of the starting point** Remember to make a note of the start and end points of your keyboard entries. The content of the key macro cannot be documented. You cannot check your entries afterwards based on the saved key macro.
Testing and running the key macro	1. Return to the start point you noted. 2. Press the key combination ***CTRL + D***. 3. **REPLAYING KEY SEQUENCE** **MACRO NAME: @@@@@@TP.INI** 4. Type in the name 5. Press the ***Return*** key. **Note** **Continuing after a message** Once the PG has output a message relating to the key macro, you can only continue working after entering a file name.

A.8 Programming Rules

This section describes some of the programming rules for changing between the methods of representationm LAD, CSF and STL. A program block written, for example, in STL cannot always be represented as a Ladder Diagram or Control System Flowchart. This also applies when you change from one of the graphical methods of representation (LAD and CSF) to the other.

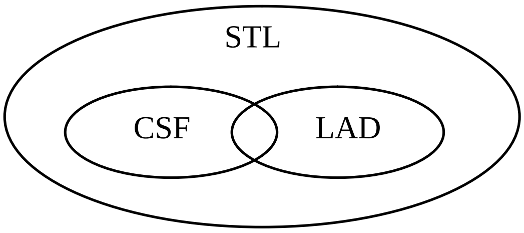


Figure A-6 Scope and Limits of the STEP 5 Methods of Representation

Note

Programs you have written in LAD or CSF can always be translated back to STL.

A.8.1 Available Blocks and Parameter Limits

Block		Parameter limits		Comment
Name	STEP5 ID	I/O on PG	Call in program	
Organisation block	OB	1–39	0–255	Max. 4096 segments per block: **length max. 4096 Kw** per block per segment 256 statements
Program block	PB	0–255	0–255	
Sequence block	SB	0–255	0–255	
Function block	FB	0–255	0–255	
Extended function block	FX	0–255	0–255	
Data block	DB	0–255	0–255	max. 2048 DW per block without header max.(6x256)+40 blocks per S5D file
Extended data block	DX	0–255	0–255	
Comment block for OB	OC	1–39	–	Size: max. 16 Kb **– max. (6 • 256) + 40 blocks per S5D file**
Comment block for PB	PC	0–255	–	
Comment block for SB	SC	0–255	–	
Comment block for FB	FC	0–255	–	
Comment block for FX	FCX	0–255	–	
Comment block for DB	DC	0–255	–	
Comment block for DX	DCX	0–255	–	
Segment comment for OB	#OBDO	0–39	–	**Size max. 16 Kb or 8 Kw** **per block** **max. 255 blocks per S5D file**
Segment comment for PB	#PBDO	0–255	–	
Segment comment for SB	#SBDO	0–255	–	
Segment comment for FB	#FBDO	0–255	–	
Segment comment for FX	#FXDO	0–255	–	
Segment comment for DB	#DBDO	0–255	–	
Segment comment for DX	#DXDO	0–255	–	
Plant comment	#Name	# + max. 8 chars.	–	
Variables block	VB	0–255	–	PLC function

- Max. size of an S5D file : 4 Mb
- –
  LAD + CSF:
  max. 400 display elements per block,
  max. 50 lines / 8 columns

A.8.2 Graphical Input in LAD and CSF

Input in LAD, Output in CSF

If you use too many nesting levels when inputting in LAD, you may exceed the display limits for output in CSF.

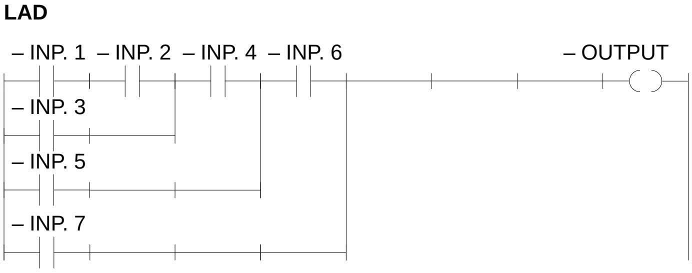


Figure A-7 Example of Nesting when Inputting in LAD

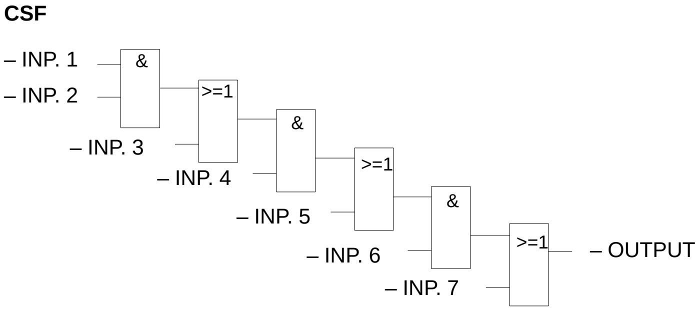


Figure A-8 Example of the Nesting above Output in CSF

Input in CSF, Output in LAD

Too many entries in a CSF box can exceed the display limits (8 levels) in LAD.

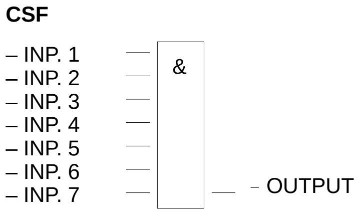


Figure A-9 Example of Nesting when Inputting in CSF

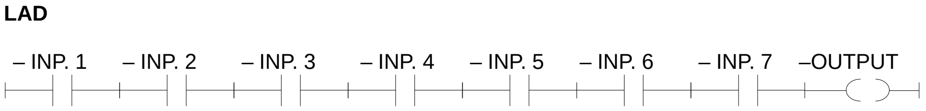


Figure A-10 Example of the Nesting above Output in LAD

Output of a Complex Element

The output of a complex element (latch, comparator, timer or counter) must not be ORed.

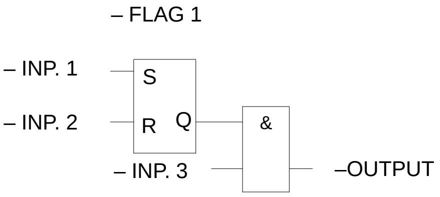
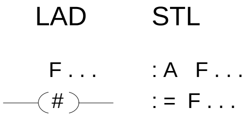


Figure A-11 Only AND Boxes are Permitted after a Complex Element

Connectors

Connectors are temporary flags used to save logic operations that recur often.
To make things clearer, the rules for connectors are listed separately for LAD and CSF. Following the rules, there is an example to illustrate both methods of representation.

- **Connectors in LAD**

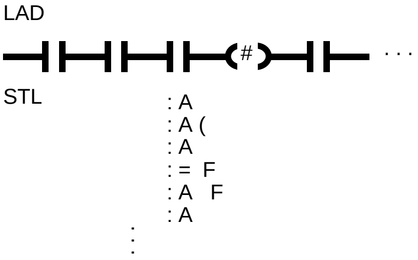


```
: A   F . . .
: =  F . . .
```

Figure A-12 Connector in LAD and STL

A connector is set to the result of the logic operation produced by the operations programmed before it on the power rail. The following rules apply:

- **Connector in series**



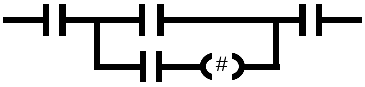
```
: A
: A (
: A
: =  F
: A   F
: A
```

Figure A-13 Connector in Series

Connectors in series with other connectors. In this case the connector is treated as a normal contact.

- **Connector in a parallel branch**

LAD



STL

```
          : A. . .
          : A (
          : A . . .
          : O (
          : A . . .
          : =  F
          : A   F
          : )
          : )
          : A . . .
       :
       :
```

Figure A-14 Connector in a Parallel Branch

In a parallel branch , a connector is treated like a normal contact. The entire parallel branch must be enclosed in parenthesis of type O (...).

A connector must never follow the power rail immediately (connector as first contact) or come directly after a power rail has been opened (connector as first contact in a parallel branch).

- **Connector in CSF**

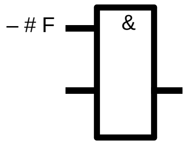
```
  CSF          STL

               : =  F . . .
– # F. . . –
               : U   F . . .
```

Figure A-15 Connector in CSF and STL

The connector is set to the result of the logic operation as a temporary flag for the entire binary logic operation before the connector. The following rules apply:

- **Connector at the first input of an AND or OR box**

CSF



STL

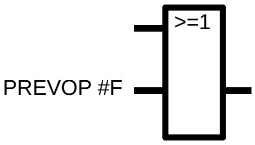
```
: =  F . . .
: A   F . . .
: A . . .
: A . . .
```

Figure A-16 Connector at the First Input

The connector is not within parenthesis.

- **Connector not at the first input of an OR box**

CSF



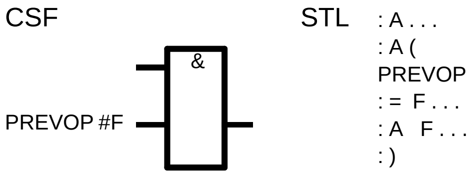
STL

```
: O . . .
: O (
PREVOP
: =  F . . .
: A  F . . .
: )
```

Figure A-17 Connector not at the First Input

The binary logic operation before the input is enclosed in parentheses of the type O (...).

- **Connector not at the first input of an AND box**



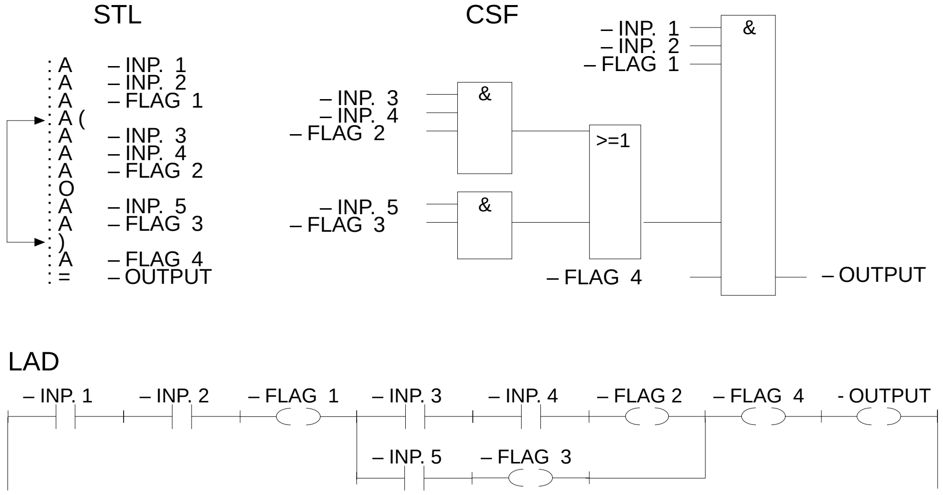
```
: A . . .
: A (
PREVOP
: =  F . . .
: A   F . . .
: )
```

Figure A-18 Connector not at the First Input

The binary logic operation before the input is enclosed in parenthesis of the type A (...).
Only allowed with CSF, this cannot be represented graphically in LAD !
(in the figures: PREVOP = previous logic operation)

Examples of Connectors

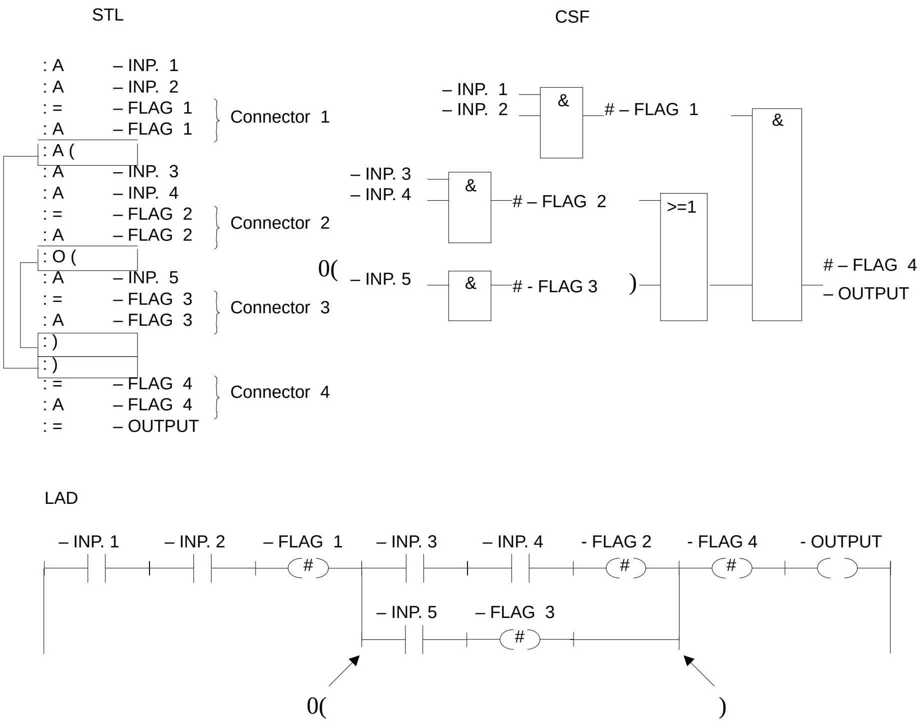
- Example without connectors



```
: A      – INP.  1
: A      – INP.  2
: A      – FLAG  1
: A (
: A      – INP.  3
: A      – INP.  4
: A      – FLAG  2
: O
: A      – INP.  5
: A      – FLAG  3
: )
: A      – FLAG  4
: =      – OUTPUT
```

Figure A-19 Example without Connectors

Example with connectors



STL

```
: A      – INP.  1
: A      – INP.  2
: =      – FLAG  1  } Connector  1
: A      – FLAG  1  }
: A (
: A      – INP.  3
: A      – INP.  4
: =      – FLAG  2  } Connector  2
: A      – FLAG  2  }
: O (
: A      – INP.  5
: =      – FLAG  3  } Connector  3
: A      – FLAG  3  }
: )
: )
: =      – FLAG  4  } Connector  4
: A      – FLAG  4  }
: =      – OUTPUT
```

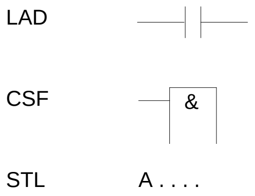
Figure A-20 Example with Connectors

A.8.3 Input in STL

You must keep to the programming rules if you want to translate the program to LAD or CSF. If you have not kept to the rules and attempt to make corrections when outputting in LAD or CSF, errors can occur when you save the program without the PG displaying an error message.

AND Operation

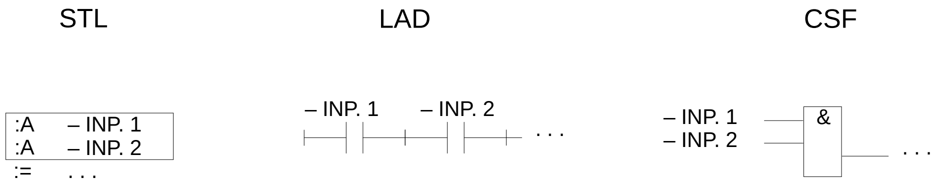
With AND operations, the operands are connected in series, the signal states of the A or AN operations are scanned and ANDed.



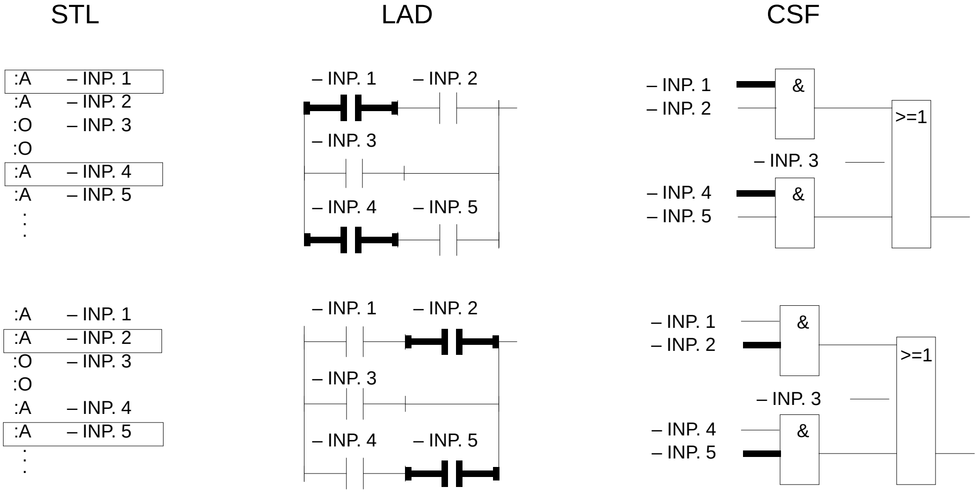
LAD: contact in series

CSF: input of an AND box

STL: statement A...



```
:A    – INP. 1
:A    – INP. 2
:=    . . .
```



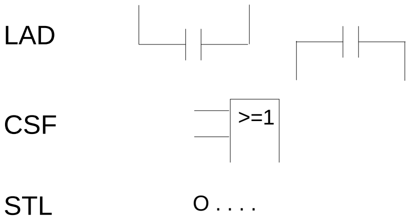
```
:A     – INP. 1
:A     – INP. 2
:O     – INP. 3
:O
:A     – INP. 4
:A     – INP. 5
 :
```

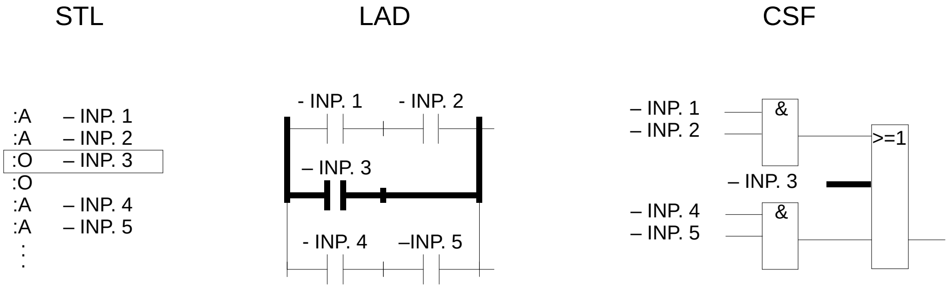
```
:A     – INP. 1
:A     – INP. 2
:O     – INP. 3
:O
:A     – INP. 4
:A     – INP. 5
 :
```

Figure A-21 Example of the Rule for AND Operations

OR Operation

Scan the signal state and perform an OR operation.
LAD: only one contact in a parallel branch
CSF: input of an OR box
STL: statement O...

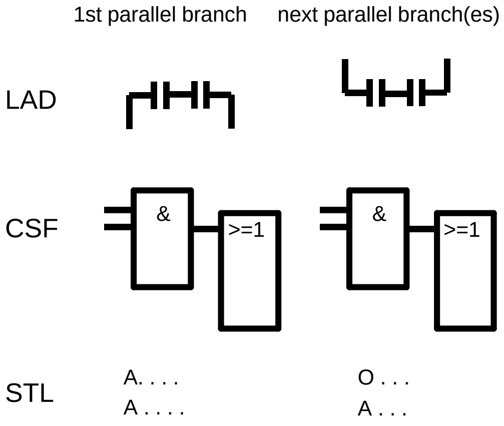




```
:A    – INP. 1
:A    – INP. 2
:O    – INP. 3
:O
:A    – INP. 4
:A    – INP. 5
 :
```

Figure A-22 Example of the Rule for OR Operations

AND before OR Operation



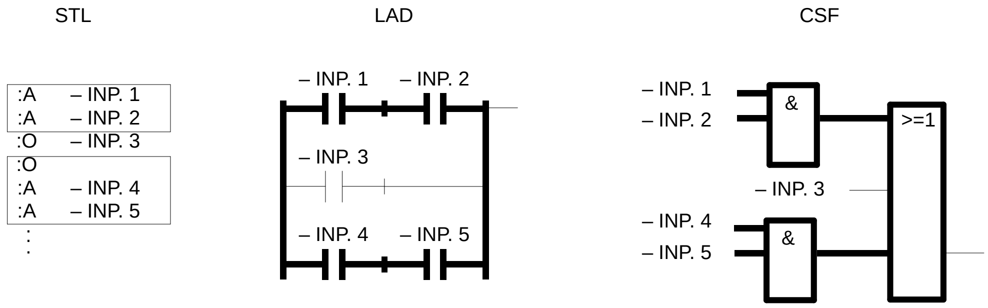
LAD: more contacts in a parallel branch

CSF: AND box before OR box

STL: statements O...

parallel branch A...
A...

STL LAD CSF

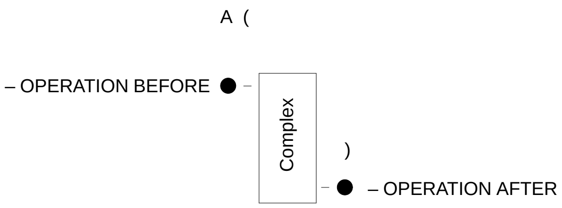


```
:A    – INP. 1
:A    – INP. 2
:O    – INP. 3
:O
:A    – INP. 4
:A    – INP. 5
 :
```

Figure A-23 Example of the Rule for an AND before OR operation

Parenthesis

This rule covers the use of parenthesis with complex, self-contained binary logic operations and complex elements with operations before and after them.



Complex binary operation
These operations include OR before AND operations.

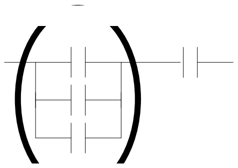
OR before AND operation

```
STL         A(
            O . . .
            O . . . .
            O . . .
            )
            A . . .
```

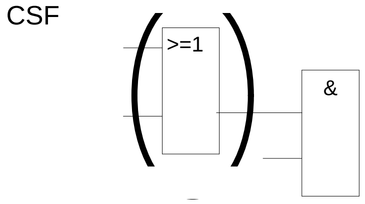
STL: statements

A(
OR operation
)
A

LAD



LAD: Connect parallel contacts in series.



CSF: OR box before AND box.

These operations are a subset of the complex binary operations, two parallel contacts being the simplest operation.

Complex Elements (latch, timer, comparator and counter functions)

The following rules apply to complex elements:

- no following operation: no parenthesis
- AND operation follows: A (...).
- OR operation follows: O (...), only for CSF.
- Complex elements cannot be followed by other operations.

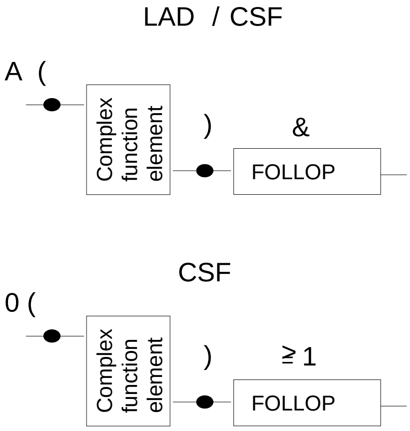


Figure A-24 Parenthesis with Complex Elements

Comparator function

A comparison of floating point numbers is only possible in STL.

Complex Elements, Undefined Inputs and Outputs

Each undefined input or output must be supplied with NOP 0 in STL.
Only one complex element is permitted per segment.

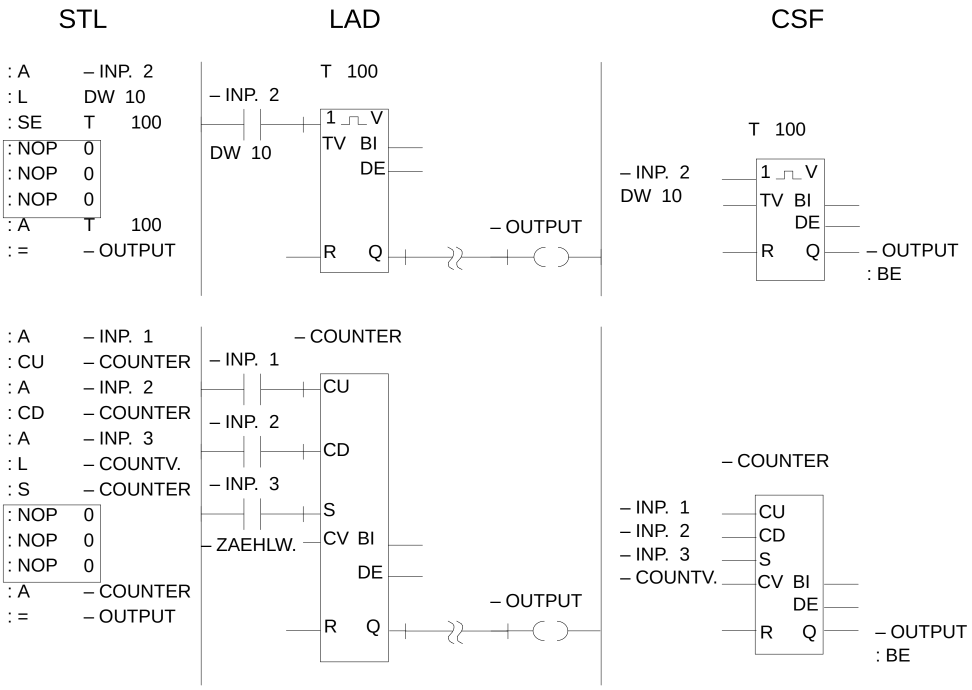


Figure A-25 Example of Undefined Inputs and Outputs in STL, LAD and CSF

Index

An
Siemens AG
AUT E 146
Östliche Rheinbrückenstr. 50
D-76181 Karlsruhe

From:
Your Name: ____
Your Title: ____
Company Name: ____________________________________
Street: ____
City: ____
Phone: ____

Please check the industry that applies to you:

❒ Automotive
❒ Chemical
❒ Electrical Machinery
❒ Food
❒ Instrument and Control
❒ Nonelectrical Machinery
❒ Petrochemical
❒ Pharmaceutical
❒ Plastic
❒ Pulp and Paper
❒ Textile
❒ Transportation
❒ Other ______________

Remarks Form

Your comments and recommendations will help us to improve the quality and usefulness of our publications. Please take the first available opportunity to fill out this questionnaire and return it to Siemens.

Please give each of the following questions your own personal mark within the range from 1 (very good) to 5 (poor).

Title of Your Manual: . . . .

Order No. of Your Manual: . . . .

1. Does the contents meet your requirements? □
2. Is the information easy to find? □
3. Is the text easy to understand? □
4. Does the level of technical detail meet your requirements? □
5. Please rate the quality of the graphics/tables: □

Additional Comments:

_ _

_ _

_ _

_ _

_ _

_ _

_ _

_ _

_ _