SIEMENS

SIMATIC

STEP 5/ST V7.0

Manual

This manual has the order number:

6ES5998-0MA24

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:

Danger

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

Warning

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

Caution

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

Note

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

Qualified Personnel

The device/system may only be set up and operated in conjunction with this manual.

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 ▶, SIMTIC NET ▶ and SIMATIC HMI ▶ are registered trademarks of SIEMENS AG.

Third parties using for their own purposes any other names in this document which refer to trademarks might infringe upon the rights of the trademark owners.

Siemens AG
Bereich Automatisierungs- und Antriebstechnik
Geschaeftsgebiet Industrie-Automatisierungssysteme
Postfach 4848, D-90327 Nuernberg

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.

Important Information

Purpose of the Manual

This manual has the following aims:

- To explain the basic concepts of the standard software
- To introduce its most important functions

The software used to configure and program the SIMATIC S5 programmable logic controllers was developed according to modern ergonomic principles. Handling the software is therefore easy to learn and to a large extent self-explanatory.

When procedures are explained, you will find the relevant menu commands are also described. However, instructions on how to fill out dialog boxes are not included since this is explained in online help.

Audience

This manual is intended for installation personnel, programmers, and service personnel who have little or no experience of working with the software package STEP 5/ST version 7.0.

Scope of the Manual

This manual is valid for the STEP 5/ST programming software, version 7.0. It is valid for the STEP 5 Standard software package and is the basis for the optional software packages.

Standards

The STEP 5 software complies with the International Electrotechnical Commission’s standard IEC 1131-3 (or EN 61131-3) for programming languages used with programmable controllers.

Installation and Authorization of the Software

Installing the STEP 5 software and transferring the authorization to hard disk is described in this manual. Please refer to Chapter 3 or the readme file for detailed information.

Structure of the Manual

This manual is divided into the following parts:

- Part 1 contains general information on terminology, basic handling of the standard STEP 5/ST software, and on preparing for a programming session. You should read the first four chapters before you start working with the software.
- Part 2 describes how to work with the language editors.
- Part 3 describes testing, handling and documenting projects.
- Part 4 describes working with special SIMATIC S5 programs.
- To familiarize you with STEP 5/ST more quickly and to illustrate a

practical application, Part 5 contains a sample application. Based on the task of controlling a carwash, the sample project guides you step by step through editing, testing, documenting, and archiving a user program.

- Part 6 introduces you to data managment within STEP 5/ST.

If you have already created a small project and gained some experience, you can read each chapter separately as and when you require information on the topic it covers.

Conventions

References to other manuals are shown as reference numbers between slashes /.../. Using these numbers you can check the exact title of the manual in the list of references at the end of this manual.

Online Help

In addition to the manual, detailed information is also available to you in the integrated online help system when you are working with the software. You can call up the help system by pressing the **F7** and **F8** keys.

Additional Assistance

If you have any questions about the software described in this manual and cannot find an answer here or in the online help, please contact the Siemens representative in your area. You will find a list of addresses in the catalogs and in Compuserve (`go autforum`).

You can contact our **SIMATIC Customer Support** by phone at the number +49 (911) 895-7000 or by fax at +49 (911) 895-7002. You can also send questions by email on the Internet or by email to the mailbox listed above.

If you have any questions or comments on this manual, please fill out the remarks form at the end of the manual and return it to the address shown on the form. We would be grateful if you could also take the time to answer the questions giving your personal opinion of the *manual*.

Siemens also offers a number of training courses to introduce you to the SIMATIC S5 automation system. Please contact your regional training center or the central training center in Nuremberg, Germany for details:

D-90327 Nuremberg, Tel. (+49) (911) 895 3154.

Information Updates

The latest information about SIMATIC products is always available:

- on the Internet under `http://www.ad.siemens.de/simatic`
- at the fax polling no. 08765-93 02 77 95 00

Our SIMATIC Customer Support can also help you with up-to-date information and downloads that can be useful when working with SIMATIC products:

- on the Internet under `http://www.ad.siemens.de/support/html–00/`
- in the SIMATIC Customer Support mailbox at the number +49 (911) 895-7100

To contact the mailbox, use a modem with up to V.34 (28.8 Kbaud), with the following parameters: 8, N, 1, ANSI, or dial on ISDN (x.75, 64 Bbits).

Contents

Part 1: Preparing for Programming

Product Overview

1

1.1 Contents of the STEP 5/ST V7.0 Package

Overview

The exact content of your system software is listed in the Product Information which is supplied either with your new PG or with your STEP 5 products.

Software Supplied with the PG

In the PG 740, for example, the **system software** Windows 95, STEP 5 V7 etc. is installed on the hard disk in compressed form in five languages. The advantage of this for you is that you can use your STEP 5 software on the PG after making only a few basic settings and selections.

1.2 New Features Compared with Version 6.x

Overview

With the introduction of STEP 5/ST V7.0, there is now an identical software package for programming devices and PCs. After installation on either type of system, STEP 5/ST cannot be copied.

Extended Functions

STEP 5/ST V7.0 has the following extended or modified functions.

- DOS shell call within STEP 5/ST: DOS commands can be called without closing STEP 5.
- Operation on a server. The authorization must, nevertheless, be installed on the local destination device.
- The STL editor/batch compiler is shipped with STEP 5/ST.
- Language change within STEP 5.
- History (repeat function) for input fields in the dialog boxes and when selecting the project.
- Extended syntax for the block list (uniform for all functions such as editor and printer).
- If you select the option in the project settings, the last active optional package is noted and is started automatically when STEP 5 is started.
- Recording of the five project files last used in the menu.
- Extension of the help and information functions in the menu and dialog boxes.
- Integration of color settings in the menu.

- Inclusion of the COM packages in the menu. The COM packages can be included in the “Change” menu allowing them to be started directly as already usual in GRAPH 5/II.
- It is now possible to print blocks with calls for missing FBs and to print ranges of blocks.

Operator Ergonomics

The ergonomics of the package have been improved:

- “Flat” menu structures: in most cases, now only two levels.
- New acceleration keys and hot keys: ***SHIFT***, ***CTRL*** and ***ALT*** supported in menus and in dialog boxes.
- Harmonization of the dialog boxes: uniform structure in dialog boxes, divided into source, destination, output, and options.
- Project settings now in the form of tab pages: simple, extended, and clear design of the project settings arranged according to the topics: PLC, Blocks, Symbols, Documentation, Eprom, and Options.

Performance

The performance has been significantly improved:

- Direct, fast editor calls: the block editors can be called directly from the project settings, the DIR and the ISTACK.
- Integration of the EPROM driver in the STEP 5/ST standard package: the EPROM driver must no longer be loaded as a resident program in memory when the PC/PG is booted.
- Revision of the editor for bus selection
- Improved installation tool
- Reduction of memory requirements (conventional RAM)
- Increased performance with reduced package loading times

Modern DOS Architecture

The architecture has been revised:

- Real DOS application
- Use of DOS paths
- No limitation on the use of drives (not for old COM packages)
- Use of extended memory (XMS and EMS)

Installing STEP 5

2

Overview

This chapter is intended to help you in the following situations:

- When installing the STEP 5 Basic Package
- When creating your working environment
- When transferring the authorization to and from the hard disk
- When installing the STEP 5 hardware
- When working with COM packages
- If you have problems involving compatibility

Chapter Overview

2.1 Software Requirements for Installation

Operating Systems

To use STEP 5/ST V 7.0, you require at least the **MS-DOS 5.0 operating system**. You can also use the following:

- Higher MS-DOS versions,
- Windows 3.x and
- Windows95

Windows NT is not supported.

CONFIG.SYS

The **CONFIG.SYS** file should include the following commands and parameters for STEP 5/ST, V7.0:

```
DEVICE     =          C:\DOS\HIMEM.SYS
DOS        =          HIGH,UMB
DEVICE     =          C:\DOS\EMM386.EXE RAM NOEMS
FILES      =          32 (must be ≥ 32)
BUFFERS    =          20
```

Windows 95 SYSTEM.INI

When using Windows 95, the following entry is possible in the **SYSTEM.INI** file in the section **[386Enh]**:

```
LocalLoadHigh=True
```

You should only use this entry if you have memory problems with COM packages. This parameter has no influence on the STEP 5 basic package. This entry is not made automatically by the installation program since this can cause problems when booting the system.

STEP5/ST V 7.0 is started with the S5.BAT call in the home directory (with a standard installation `DR:\STEP5\S5_HOME`).

AUTOEXEC.BAT

The **AUTOEXEC.BAT** file should include the following commands and parameters for STEP 5/ST V7.0:

```
PATH C:\;C:\DOS;C:\STEP5\S5_HOME;
SET S5_HOME= C:\STEP5\S5_HOME
SET S5_SYSTEM= C:\STEP5\S5_SYS
```

These entries are made by the installation program.

By including the STEP 5 directory in the PATH statement in AUTOEXEC.BAT, it is possible to start STEP 5 from any directory.

An entry in AUTOEXEC.BAT allows SIMATIC NET network drivers to be called automatically. The entry is made when you call S5DRV.BAT.

2.2 Hardware Requirements for Installation

Overview

STEP 5/ST V 7.0 for PCs can be installed on PC/AT 03 compatible devices. The minimum memory requirements are 3 Mbytes of XMS memory = 4 Mbytes RAM.

If you use the P Tools any disk cache controller present must be deactivated.

2.3 INSTALL Installation Program

Brief Overview

The package is installed by the Install.exe program, simply called INSTALL below. INSTALL is a menu-guided MS-DOS program and can therefore be run under Windows 3.x and Windows 95 in a DOS box.

2.3.1 Before Starting Install

Basic Requirements

Your programming device must have a drive C:. INSTALL creates the directories:

`C:\S5_INFO` and
`C:\SINEC`.

INSTALL also expects to find the configuration files

`C:\CONFIG.SYS` and
`C:\AUTOEXEC.BAT` on the drive.

If the file `C:\AUTOEXEC.BAT` does not exist, INSTALL will create it.

To run INSTALL, you require at least 400 Kbytes of free MS-DOS memory on your programming device.

The installation program requires the environment variable TMP. Under Windows 95, the variable is set automatically. Under Windows 3.x and MS-DOS, the variable should be set in the configuration file `C:\AUTOEXEC.BAT` to an existing directory for temporary files
(for example SET TMP=C:\TMP).

To represent semi-graphic characters correctly, it is advisable to configure an ANSI driver on your programming device. Please check whether the ANSI driver is loaded in your configuration file `CONFIG.SYS`

`DEVICE=ANSI.SYS`.

When you start in the DOS box in Windows, the screen setting should be set to full screen to avoid distorting the display of the program. This is included in the supplied file INSTALL.PIF.

Before you start a server installation, make sure that the read-only attribute of an existing STEP 5/ST system directory with the same directory name on your server is canceled using the tools provided by your operating system. Once the server installation is completed, you can set the read-only attribute again.

2.3.2 Working with the User Interface

User Interface

INSTALL can be controlled with either the mouse or keyboard. Please refer to the summary of keyboard operations in Section 2.8.6.

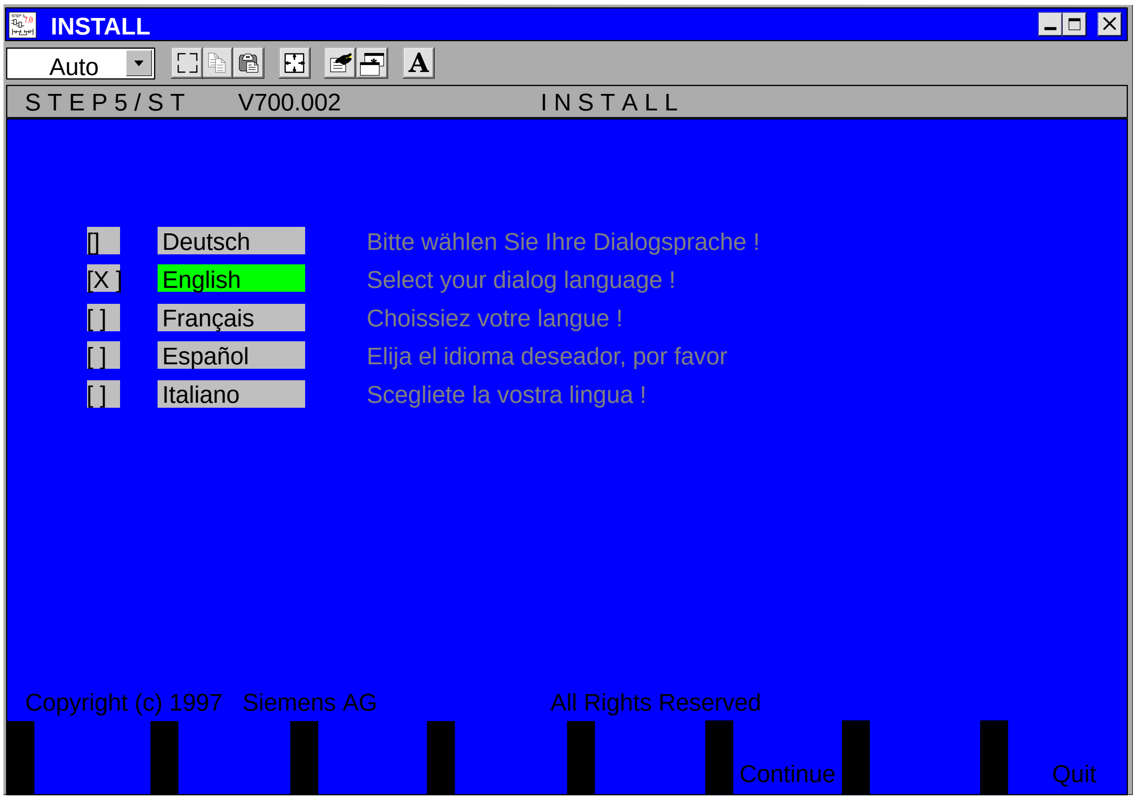


Figure 2-1 Language Selection Dialog

2.3.3 Starting Installation

Ways of Starting the Installation

To start the installation, insert the diskette STEP 5/ST V7.0 Disk #1 in a 3.5” drive (for example drive A:), change to drive A: and start INSTALL.

Table 2-1 Installation with the Various Operating Systems

Operating System	Steps
Starting under DOS	1. A: 2. INSTALL
Starting under WINDOWS 3.x	1. Start File Manager 2. Select A: 3. Double-click INSTALL.EXE
Starting under WIN95	1. Start Explorer 2. Select A: 3. Double-click INSTALL.EXE
Starting from CD	1. Select CD-ROM drive 2. Select STEP5.INS directory 3. Type in INSTALL

INSTALL automatically detects the operating system under which it is started.

Call Syntax

The complete call syntax for the installation is as follows:

```
INSTALL [/h<path>] [/s<path>] [/m] [/c] [/?]
```

Table 2-2 Command Parameters of INSTALL

Parameter	Description
`[/h<path>]`	With this parameter, you can select the directory for the STEP 5/ST home directory. The default directory \STEP5 is then overwritten. The home directory is assigned the directory name \S5_HOME internally.
`[/s<path>]`	With this parameter, you can select the directory for the STEP 5/ST system directory. The default directory \STEP5 is overwritten. The system directory is assigned the directory name \S5_SYS internally.
`[/m]`	Use this parameter if your programming device is connected to a monochrome monitor. INSTALL then provides menus optimized for black and white display. If there is already a STEP 5/ST home directory with the same name as the default name used by INSTALL, and if this contains the file @@@@@@FT.DAT, INSTALL will automatically be started in black and white. This does not apply if you select the parameter /c.
`[/c]`	This is the default for the installation program. Please refer to the paragraph above about the parameter /m.
`[/?]`	With this parameter, you obtain a brief overview of all the selectable start parameters.

2.3.4 Cancelling the Installation

You can cancel the installation in the following ways:

- With the menu by selecting ***Cancel*** ▸ ***Quit***, or
- directly with the key combination ***Ctrl*** + ***C***

Please note that aborting the program with ***Ctrl + C*** can lead to an incomplete installation. If you want to work with STEP 5/ST, it is advisable to repeat the installation following this program abort.

2.3.5 Installation Log

Overview

The installation is logged. The Install.log file is created in the directory indicated by the TMP environment variable. If TMP is not configured on your programming device, the LOG file is written to C:\.

The LOG file contains all the messages from the operating system and the extractor. If problems occur in the installation, you can open INSTALL.LOG with a text editor and analyze the entries it contains.

The LOG file is overwritten by any new installation.

Display Detailed Information

After installing the software, you will find files of the type *.TXT (for example, STEP5E.TXT) or of the type *.WRI (for example PRODE.WRI) in the C:\S5_INFO\ directory. You can use a suitable editor (for example EDIT or WordPad) to read or print the files. These files contain detailed information on the individual software components.

Example: Using the MS-DOS command

```
TYPE STEP5E.TXT | MORE
```

you can display the file `STEP5E.TXT` page by page.

Help Directly on the Screen

You can enter MS-DOS commands with the option **/?** to display brief information about the function and the permitted options.

Under Windows**,** you can obtain help on the current function at any time by pressing the ***F1*** *key.* You can also obtain information from the entries in the ***Help*** menu.

On PGs, you can also display help texts on the current function while working in STEP 5 by pressing ***SHIFT*** + ***F8*** or the ***Help*** key (PG 730/750/770).

This ***Help*** key has been replaced on the PG 720 and PG 740 by the keys F_n + ***F1***.

2.4 Installation Procedure

Steps in Installation

Installing your STEP5/ST software involves several steps.

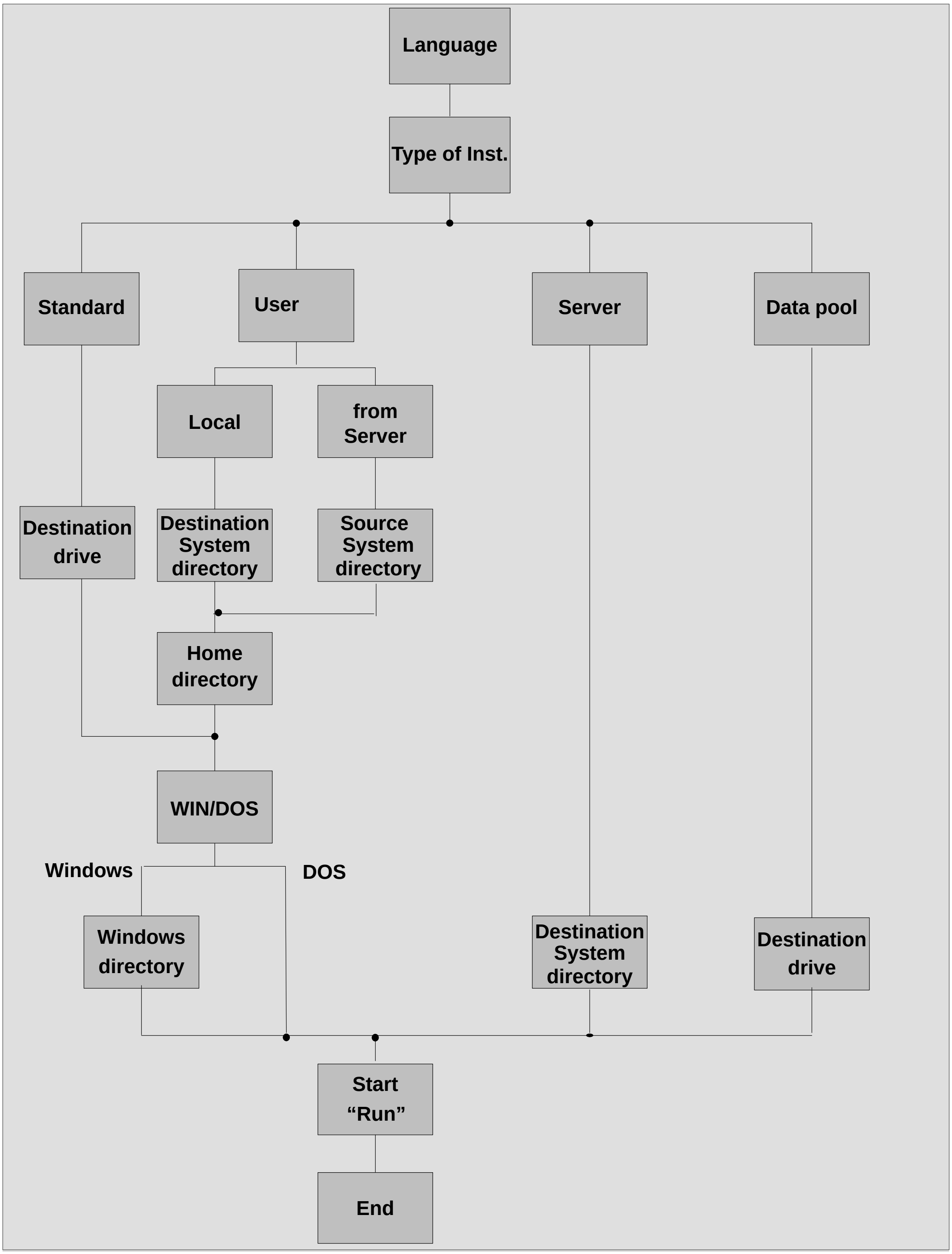


Figure 2-2 Steps in Installation

2.4.1 Selecting the Dialog Language

The installation software and the STEP 5/ST software is shipped in five languages. After you start the installation, you will be prompted to select the dialog language you require. Five languages are available in the standard package.

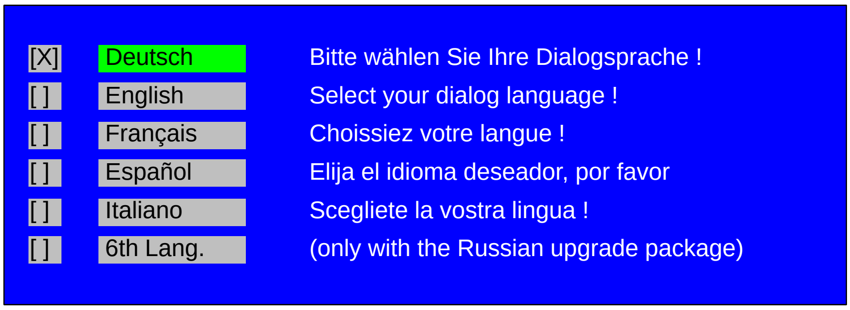


Figure 2-3 Language Selection

You can select the language with the mouse or cursor up/down keys or the tab key. INSTALL uses the configured system language on your programming device set with COUNTRY=. If there is no COUNTRY setting, English is the default language.

1. Select the required language.
2. Press ***F6*** *= Continue* to continue with the installation.

After you have selected the language, the subsequent menus and dialogs are displayed in this language. The **STEP 5/ST software** is always installed in **five languages**.

2.4.2 Types of Installation

There are four types of installation available allowing you to select the working environment you require.

By pressing the ***Ins*** key or the ***F6*** *key (Continue)* you display the basic dialog of the installation program. Here, you can decide on the type of installation. There are four types of installation available:

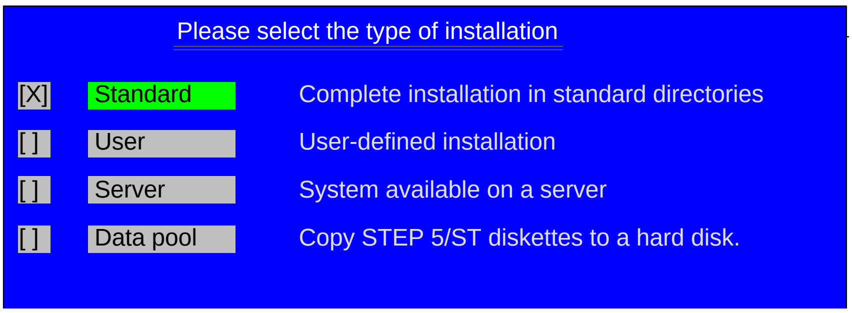


Figure 2-4 Types of Installation

2.4.3 Standard Installation

The quickest method of obtaining a completely installed version of STEP 5/ST on your programming device is to select the *Standard* Installation.

In the standard installation, the directory names for the STEP 5/ST data are preset and cannot be modified by the user.

The following directory names are preset:

Directory	Path	Content
System Directory	C:\STEP5\S5_SYS	STEP 5 /ST system data
Home Directory	C:\STEP5\S5_HOME	STEP 5 /ST device data

The only modification available to the user is the drive.

Changing the Destination Drive

In the standard installation you are first prompted to confirm or modify the default destination drive for the STEP 5/ST data.

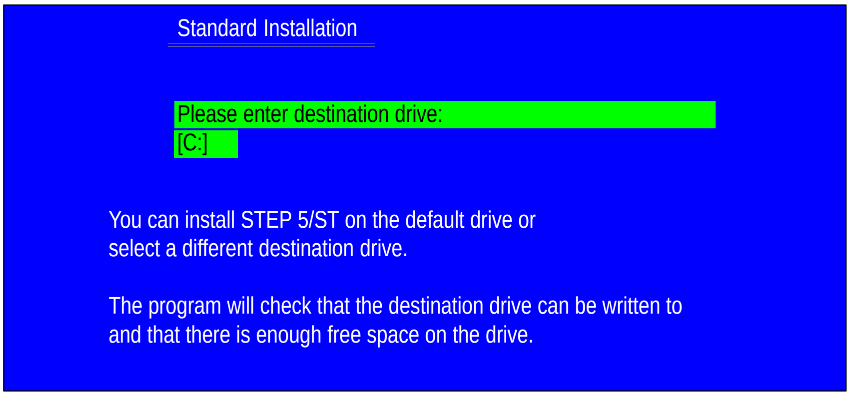


Figure 2-5 Drive Selection for Standard Installation

Before you exit the dialog, the drive you have selected is checked to make sure that it is suitable for the STEP 5/ST installation. The drive must meet the following criteria:

- It must exist and be ready (for example exchangeable data medium inserted)
- It must be read/write (not a compact disc)
- It must have sufficient free hard disk space.

If any of these criteria is not met, error messages will be displayed. In this case, either select a different drive or correct the problem and then restart the installation.

Note

Selecting a drive higher than P: is not advised. Although STEP 5/ST can work with any drive, drives higher than P: are not supported by all COMs (STEP 5/ST menu item: *Change*...). If you have selected a drive higher than P: and potential problems are recognized, a warning is displayed.

Windows Integration

If you make the installation in a DOS box in the Windows 3.x or Windows 95 operating system, Install displays options for Windows integration in the following two dialogs. If you work under MS-DOS and have not started Windows, these dialogs are not displayed.

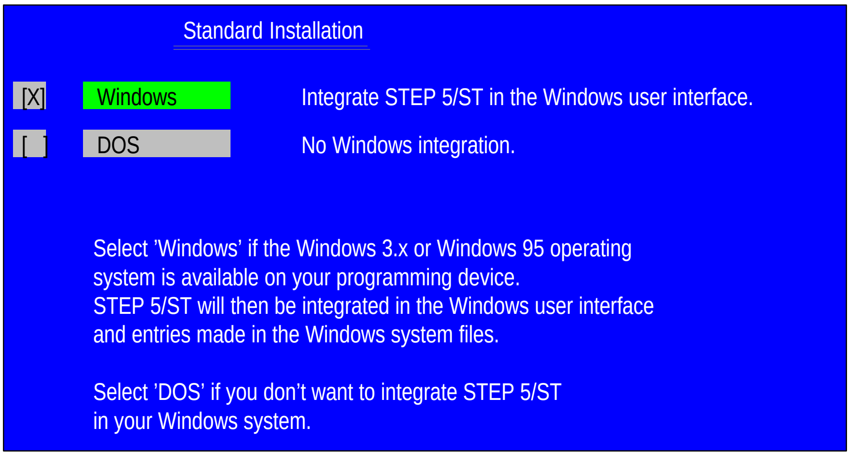


Figure 2-6 Windows Integration

In the first dialog, you are asked whether you want to integrate the STEP 5/ST program start icons in the Windows user interface. If you select *Windows,* the following actions are executed after starting the program:

Working with the **Windows 95** operating system

- Your STEP 5/ST programs are added to the Windows start menu in the *Simatic\Step5v70* submenu. The PIF files belonging to the programs are copied from the STEP 5/ST system directory to the Windows start menu directory and adapted to the current Windows, STEP 5/ST system and home directory names.
- Any existing *Step5v70* start menu is first copied to *Step5v70.~xx*.

Working with the **Windows 3.x** operating system

- Your STEP 5/ST programs are added to a new program group *STEP 5 V7.0* in the Windows Program Manager. The PIF files belonging to the programs are copied from the STEP 5/ST system directory to the Windows directory and adapted to the current Windows, STEP 5/ST system and home directory names.
- STEP 5/ST system file *Step5v70.grp* is saved as *Step5v70. ~xx* and the new version transferred from the STEP 5/ST system directory.
- The Windows system file *Progman.ini* is saved as *Progman. ~xx* and then extended by the new menu entry Step5v70.

Note

Under Windows 3.x, the group file for the STEP 5/ST program start icons is only initialized for the standard directory setting C:\STEP5\....

Adapting Group File STEP5V70.GRP

The group file *STEP5V70.GRP* cannot be adapted by INSTALL. If you have changed the destination drive or directory names, you must adapt the group file *STEP5V70.GRP* in the Windows 3.x Program Manager on completion of the installation.

Select the menu option **File** ▶ **Submenu** ▶ **New...** to create a new program group and specify STEP 5 V7.0 as the description and the following file name as the group file with the corresponding Windows directory name:

`DR:\Windows\STEP5V70.GRP`
(For `DR:\Windows` specify your current Windows directory).

After you confirm these entries, the group STEP 5 V7.0 with the STEP 5/ST program start icons appear in the window of the Program Manager.

If you select DOS, these actions are not executed.

Entering the Windows Directory

In the second dialog, you are asked to confirm or enter a Windows directory. INSTALL has already examined the configuration of your programming device and proposes the current Windows directory. This proposal is taken from the WINDIR environment variable. If the variable is not set, C:\Windows is proposed.

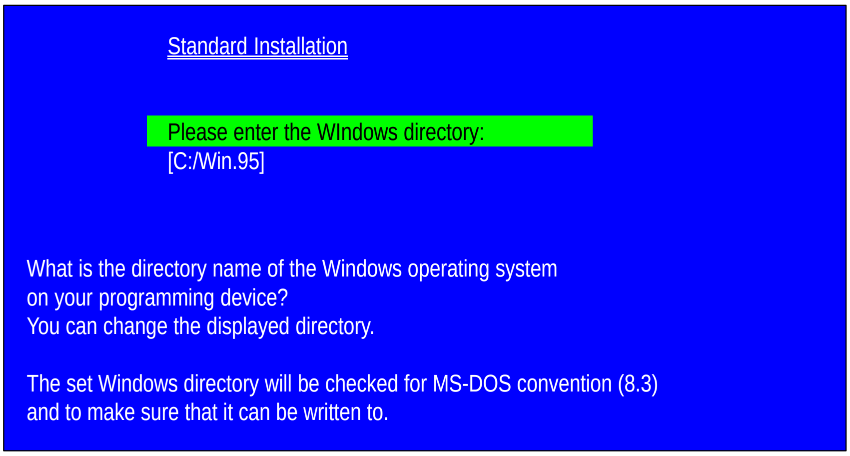


Figure 2-7 Windows Integration

The Windows directory you select is not only subjected to the syntax check described in the section *Edit Directory Names* but the following criteria are also checked when you exit the dialog:

- The directory must exist.
- It must contain the file WIN.COM.

If these criteria are not met, error messages are displayed. In this case select either a valid Windows directory or return to a dialog and then select the DOS menu item to suppress Windows integration.

Starting the Installation

In the *Start Standard Installation* dialog, you obtain a summary overview of the type of installation you have selected.

The directories are displayed in which the installation will be made after you press the function key ***F6*** *= Continue*.

Start – Standard Installation

You are installing in the following directories:

System	= C:\STEP5\S5_SYS
Home	= C:\STEP5\S5_HOME
SINEC	= C:\SINEC\BIN
Info	= C:\S5_INFO
Windows	= C:\WIN.95

The configuration files AUTOEXEC.BAT and CONFIG.SYS,
and PROGMAN.INI, STEP5.GRP and S5*.PIF
will have the information required for STEP 5/ST added to them.

All the configuration and home files will be saved as backup copies.
For more information press the <F1> function key.

Figure 2-8 Starting Installation

During the system installation, the STEP 5/ST archive names currently being unpacked are displayed in a window *Installation active ...*. During the home installation, the message *Installation active, please wait* is displayed.

The STEP 5/ST archives are compressed and are self-extracting. The following error messages can be sent by the extractor to INSTALL and displayed in an error message box on the screen:

- `DISK_FULL,`
- `CTRL_C_ABORT,`
- the group error message `ZIP_ERROR.`

If an extractor error occurs, the cause of the error must be eliminated and the installation restarted. When locating the cause of an error, it is often useful to open the log file Install.log with a text editor and to analyze the contents (see 2.3.5).

When you install the STEP 5/ST package from diskettes, once a diskette has been installed, you will be prompted to insert the next diskette. INSTALL checks whether or not you have inserted the required diskette. If not, INSTALL prompts you to insert the required diskette.

Completion of Installation

On completion of the installation, a completion dialog is displayed.

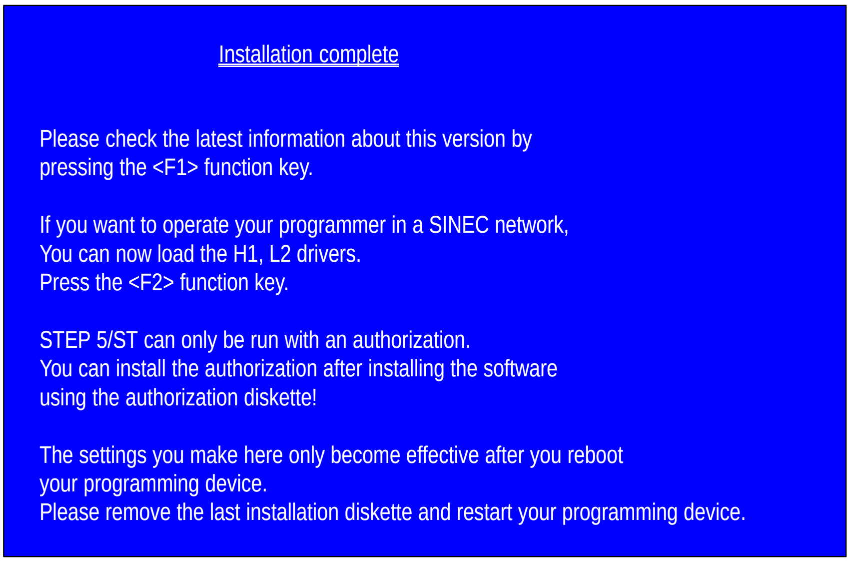


Figure 2-9 Installation Completed

What to Do after Installation

With the **F1** key, you can now read the latest information about this version of STEP 5/ST. INSTALL calls the text file STEP5x.TXT (x stands for the language version).

You can scroll line by line or page by page, forwards and backwards through the text.

You can print either the current screen page or the entire text file on your current printer.

Following a complete or home installation, you can load the required H1 or L2 DOS drivers for a SINEC network by pressing the ***F2*** key. INSTALL then changes to the STEP 5/ST MS-DOS driver load program S5DRV that can also be called separately.

2.4.4 User Installation

In the user installation, the first variable part of the directory name (for example C:\Step5) can be selected for the STEP 5/ST data. The second, fixed part of the directory name (\S5_SYS and \S5_HOME) is added by INSTALL.

When you first install STEP 5/ST V7.x, the following directory names are set as defaults:

Directory	Path	Content
System directory	`C:\STEP5\S5_SYS`	STEP 5 /ST system data
Home directory	`C:\STEP5\S5_HOME`	STEP 5 /ST device data

With each further installation, the two directory names, are taken from the STEP 5/ST environment variables S5_SYSTEM and S5_HOME set by the previous installation and set as defaults.

If you select user installation, there are two different ways in which you can install the packages. You can do the following:

- You can make a complete local installation of the system and device data
- Make a supplementary installation of the device data from a server to the local home directory

Figure 2-10 Installation Options

Local User Installation

If you have selected *Local User Installation*, you must confirm or enter a destination directory for the installation of the STEP 5/ST system data on your local programming device.

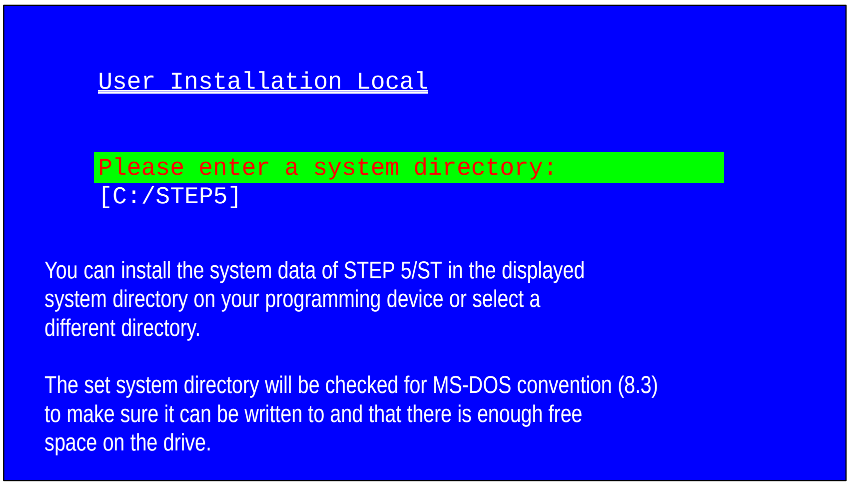


Figure 2-11 Local User Installation

User Installation From Server

If you have selected *User Installation from Server*, you confirm or enter the source directory (usually a data source on the server) in this dialog. The STEP 5/ST device data for installation on your local programming device are fetched from this directory.

Select *from Server*, if you only want to install the STEP 5/ST device data in a home directory on your programming device. In this case, the system data must already exist.

Generally, the STEP 5/ST system data are available on a server to which your programming device is connected via a network (see section *Server Installation*).

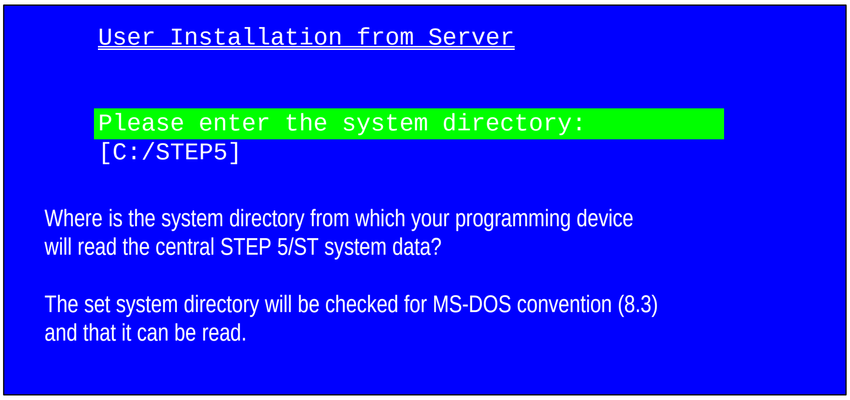


Figure 2-12 User Installation from Server (System Directory)

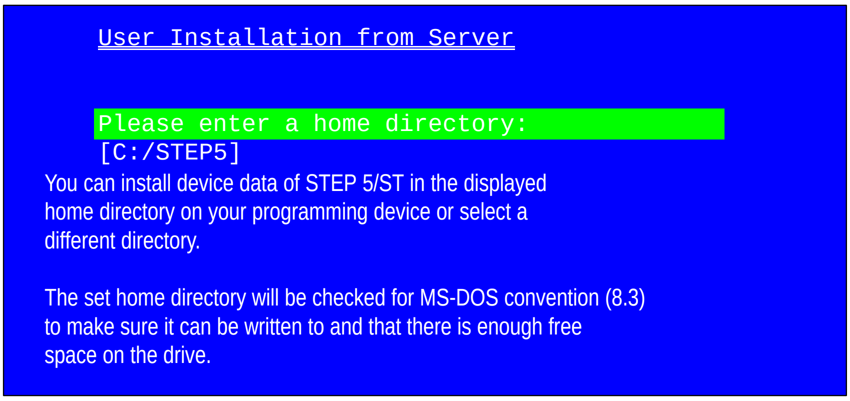


Figure 2-13 User Installation from Server (Home Directory)

The variable part of the system directory name and the home directory can have different names. The data destination of the two directory names can be on different drives on your programming device.

In this case, please remember that when you reference the system directory you must specify a data source.

2.4.5 Server Installation

Use this menu item if you want to keep the STEP 5/ST system data centrally for more than one user on a server in your network.

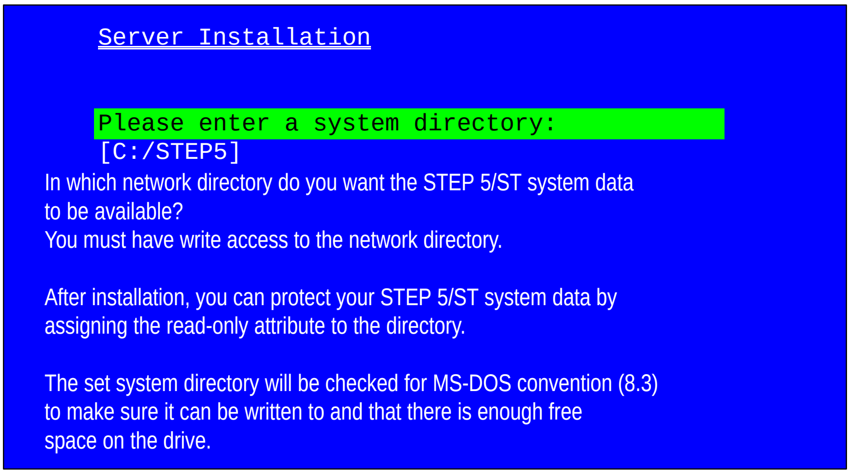


Figure 2-14 Server Installation

The first, variable part of the directory name (for example C:\Step5) can be changed to any other name for the STEP 5/ST system data. The second, fixed part of the name (\S5_SYS) is added by INSTALL.

The following directory name is set as default by INSTALL:

Directory	Path	Content
System directory	`C:\STEP5\S5_SYS`	STEP 5 /ST system data

If you start the server installation in the network from your local programming device and the S5_SYSTEM environment variable has already been set by a previous installation on this device, the content of this environment variable is proposed.

You can confirm the proposed directory or enter a different directory.

Note

Please remember, that if you select the server installation, no device data are installed. The STEP 5/ST home directory is not created. The configuration files CONGIG.SYS and AUTOEXEC.BAT are not adapted. There is no Windows integration.

Home Installation

You can only call STEP 5/ST when you install the device data extra on your programming device. To do this start with the home installation as follows:

1. On completion of the server installation, start INSTALL again, this time on the programming device in your network on which you want to work with STEP 5/ST.
2. Select the menu item *User* and *from Server*.
3. Enter the system directory name of the previous server installation and then continue with the home installation.

You can repeat the home installation on any number of programming devices in your network.

Note

To protect your STEP 5/ST system data from unauthorized access, you can assign write protection to the system directory (exception: the S5_COM\ subdirectory must not be write protected). To make the directory read only, use the tools of your operating system after the server installation is completed. Do not forget to cancel the write-protection before repeating a server installation in the same directory.

The submenu ...S5_COM\ must **not** be write-protected!

2.4.6 Data Pool Copy

This menu item allows all the STEP 5/ST installation diskettes to be put together in one directory on a hard disk or network drive.

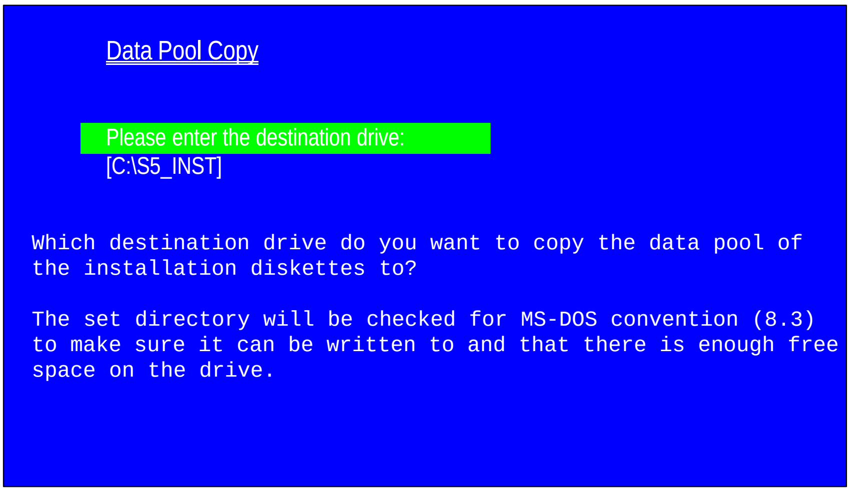


Figure 2-15 Installation Type Data Pool Copy

Please note that the Data pool option is only available when you install STEP 5/ST from diskette or from CD.

The software is compressed on the diskettes. The data and archive are stored unchanged in the selected destination directory. With a data pool, it is possible to install STEP 5/ST quickly from the local hard disk or via the network. Maintaining the STEP 5/ST data pool on a network drive makes it easier to maintain and archive.

The following directory name is proposed as default by INSTALL:

Directory	Path	Content
Data pool directory	`C:\S5_INST`	STEP 5/ST Installation data

You can confirm the proposed directory or select a different one.

The data pool directory you select, is checked for the following criteria when you exit the dialog:

- The drive must exist and be ready (for example exchangeable data medium inserted),
- The drive must have sufficient free space,
- You must have write access to the directory.

If the directory does not meet these criteria, error messages are displayed. In this case select a valid destination directory.

With a data pool, STEP 5/ST is not yet capable of execution. You can only call STEP 5/ST after running a complete installation with the menu items *Standard* or *Server*.

2.5 STEP 5/ST V 7.0 on CD-ROM

Content of the CD

The CD-ROM shipped with the package contains the following:

\STEP5.INS with all files required for the installation (data pool)

- \DISK1 content of installation diskette 1
- \DISK2 content of installation diskette 2
- \DISK3 content of installation diskette 3
- \DISK4 content of installation diskette 4
- \DISK5 content of installation diskette 5

\ACROBAT

- \DOS the Acrobat Reader for MS-DOS.
- \DISK1 Note: Acrobat Reader for MS-DOS can only
- \DISK2 be installed from diskettes
- \WIN3 the Acrobat Reader for WINDOWS 3.x and WIN95

\STEP5.DOC the *STEP 5/ST Version 7.0* manual.

2.5.1 Installing from CD-ROM

STEP 5/ST V7.0 can be installed by starting INSTALL.EXE in the \STEP5.INS directory on then CD-ROM.

Diskettes on the CD-ROM

You can create sets of diskettes by simply copying the files from the directories DISK1 to DISK4. You require 3.5” diskettes with at least 1.44 Mbytes. Format the diskettes before copying from CD.

Example: The CD drive is Drive E:

1. Type in E:CD \STEP5.INS
2. Insert an empty formatted 3.5” HD diskette
3. Type in COPY DISK1*.* A:
4. Remove the diskette and label it STEP 5/ST V 7.0 DISK 1.
5. COPY DISK2*.* A:
6. Remove the diskette and label it STEP 5/ST V 7.0 DISK 2.
7. Repeat the procedure for DISK4 and DISK5.

2.6 Special Features

2.6.1 Installing STEP 5 under MS-DOS

STEP 5/ST V7.0 can be run under MS-DOS (Version 5.0 or higher). If you run the package under MS-DOS, note the following points:

Under **MS-DOS** you can start **STEP 5/ST V 7.0** from within any directory with the **S5** call.

COM and optional packages for STEP 5/ST can only be run under MS-DOS. If you have problems with memory space, use the EMM386.EXE program in the CONFIG.SYS file to free as much memory as possible. If you are working exclusively under MS-DOS, it is possible to extend the conventional memory by 96 Kbytes. (see sample configuration).

Under MS-DOS, a printer interface can be assigned to a serial port to allow you to work with serial printers.

MS-DOS Command	Effect
MODE COM2:,9600,n,8,1,p	Setting 9600 bps, 8-bit without parity, 1 stop bit, no time monitoring
MODE LPT1:=COM1:	The printer is connected to COM1:

2.6.2 Installing STEP 5 under Windows 3.x or Windows 95

Setting up STEP 5 in Windows 3.X

If you use the Windows operating system, you can also start STEP 5 from the Windows user interface by installing STEP 5 as a Windows program. Select *Integrate STEP5/ST in the Windows User Interface* (see Section 2.4.3 *Adapting Group File STEP5 V 7.0.GRP*).

Note

Under Windows 3.x, the group file for the STEP 5/ST program start icons is created only for the standard directory setting C:\STEP5\....
When you start STEP 5 under Windows, all MS-DOS applications must be closed.
STEP 5/ST version 7.0 under Windows must only be started using the STEP 5 icons in the Program Manager in the STEP 5 window!
Starting STEP 5/ST with S5.BAT in a Windows window (for example the MS-DOS prompt) is not permitted because it can lead to a system crash due to access conflicts or to errors in online or EPROM functions.
The COM and *** keys (comment and end of segment) are not active under WIN 3.x. In this case, the corresponding function keys Fx must be used.

Recommendation

We strongly advise that you use a full screen mode when using the online functions. If you use a Window mode, this can result in considerable disturbances in communication.

2.6.3 Installing STEP 5 under Windows 95

Overview

STEP 5/ST V 7.0 can be run under Windows 95. Keep to the following rules:

- Other programs can be loaded, but must not access the resources used by STEP 5/ST (printer, EPROM ...). STEP 5/ST should only be started once.
- We strongly advise that you use a full screen mode when using the online functions. If you use a Window mode, this can result in considerable disturbances in communication.
- The Copy and Paste Windows functions must not be used for STEP 5/ST.
- If you work with online functions via the AS511, no other application should be active. This can lead to a breakdown of the online connection. It is strongly advised to work only in the full screen mode.
- Online functions via SINEC H1 / L2 are not critical if current Windows 95 drivers (VxD) are used.

Printing with Windows 95 and Networks

To print with Windows 95, the printer must be correctly configured. You must either specify that printing will be from DOS programs when you install the printer or a printer interface must be assigned later by modifying the properties of the printer.

Starting STEP 5/ST

To allow problem-free operation in STEP 5/ST under Windows 95, PIF files are specified as the default. These files are entered during installation in the Windows Start menu in a Windows 95 environment.

After installation, STEP5/ST can be called with the sequence Start menu: **Simatic** ▸ **Step5v70** ▸ **STEP 5/ST**.

The following PIF files are available:

Name	Call
STEP 5 full screen display	S5.BAT call in the full screen mode
STEP 5 window display	S5.BAT call in the window mode
STEP 5 MS-DOS mode	S5.BAT call in the MS-DOS mode with separate AUTOEXEC.BAT, CONFIG.SYS
S5 driver installation	Call S5DRV.BAT
S5 Keyboard editor	Call S5KEDIT:BAT

When using larger files and COM and optional packages, it is recommended that you start STEP 5/ST, V7.0 under MS-DOS.

Note

The Windows functions copy and paste must not be used for STEP 5/ST.

Other programs can be loaded but must not access the resources (printer, EPROM ...) used by STEP 5/ST. STEP 5/ST should only be started once.

WIN-NT

STEP 5/ST can only be operated offline under Windows NT. No EPROMs can be processed. Windows NT does not recognize the installation. There is therefore no entry in the start menu.

2.6.4 Sample Configuration

Below, you will find examples that can, if necessary, be adapted to your particular system. You can also delete or “rem” existing entries to save DOS memory.

The directories for a STEP 5/ST have the default values for the standard installation.

The lines DEVICE=...DISPLAY.SYS in the CONFIG.SYS file and the two lines MODE CON ... in AUTOEXEC.BAT can always be ”remmed” out (these apply to systems using the standard PC character set).

MS-DOS 5.0

CONFIG.SYS:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB
FILES=32
BUFFERS=20
COUNTRY=049,C:\DOS\COUNTRY.SYS
SHELL=C:\COMMAND.COM /P /E:2048
```

AUTOEXEC.BAT:

```
PROMPT $P$G
PATHHT=C:\DOS;C:\;C:\STEP5\S5_HOME
SET S5_HOME=C:\STEP5\S5_HOME
SET S5_SYSTEM=C:\STEP5\S5_SYS
LH KEYB GR,,C:\DOS\KEYBOARD.SYS
```

Optimization

If you work with MS-DOS only, in other words you do not use graphic applications, you can increase the program memory by 96 Kbytes by changing the following line (in CONFIG.SYS)

```
DEVICE=C:\DOS\EMM386.EXE NOEMS
```

to

```
DEVICE=C:\DOS\EMM386.EXE NOEMS I=A000-B7ff
```

Warning

This modification must not be made if you are using Windows 3.x or Windows 95.

WIN 3.x

WFW 3.11 makes additional entries during installation. These should not be modified manually!

MS-DOS 5.0 CONFIG.SYS:

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB
FILES=32
BUFFERS=20
COUNTRY=049,C:\DOS\COUNTRY.SYS
SHELL=C:\COMMAND.COM /P /E:2048
DEVICE=C:\WINDOWS\IFSHELP.SYS
```

AUTOEXEC.BAT:

```
SET S5_HOME=C:\STEP5\S5_HOME
SET S5_SYSTEM=C:\STEP5\S5_SYS
LH KEYB GR,,C:\DOS\KEYBOARD.SYS
```

WIN95

CONFIG.SYS:

```
DEVICE=C:\WINDOWS\HIMEM.SYS
DEVICE=C:\WINDOWS\EMM386.EXE NOEMS
DOS=HIGH,UMB
FILES=32
BUFFERS=20
COUNTRY=049,C:\WINDOWS\COUNTRY.SYS
SHELL=C:\COMMAND.COM /P /E:2048
```

AUTOEXEC.BAT:

```
PROMPT $P$G
PATH=C:\;C:\WINDOWS;C:\STEP5\S5_HOME
SET S5_HOME=C:\STEP5\S5_HOME
SET S5_SYSTEM=C:\STEP5\S5_SYS
LH KEYB GR,,C:\DOS\KEYBOARD.SYS
```

2.7 Authorization

Overview

Before you can work with your STEP 5/ST basic package or STEP 5/ST Mini PLCs, you must install an authorization that allows you to use STEP 5/ST. Without authorization, you cannot run this protected program.

2.7.1 Authorization for STEP 5/ST for PCs

STEP 5/ST includes an authorization diskette. Using this diskette, you can install or uninstall the authorization on your computer.

Installing / Uninstalling Authorization

To install or uninstall an authorization, you require your authorization diskette with the AUTHORS.EXE authorization program.

The program is self-explanatory with a comprehensive help system. The help texts are available in five languages (German, English, French, Italian, and Spanish).

All the functions of AUTHORS.EXE including the menu selection, termination and cancellation can be activated both with the mouse or with the keys displayed. You can select a function from the menu or using the function key bar at the lower edge of the screen.

Procedure

Follow the steps below:

1. Insert your authorization diskette V7.0 in drive A:
2. Type in A:\>AUTHORS and confirm with ***RETURN***.

Note

The authorization required to run STEP 5 cannot be installed on a compressed drive (for example DBLSPACE), but only on a physical drive. If STEP 5 is installed on a compressed drive, the authorization must be installed separate from the STEP 5 installation on a physical drive using the AUTHORS.EXE tool.

For STEP 5 to find the authorization on this drive, the standard entry DriveAuth = C: must be adapted in the S5_HOME\S5@@@@CF.INI file in the section *Options*.

This file can be edited with any text editor. Enter the physical drive defined with AUTHORS.EXE.

2.7.2 Authorization for UPGRADE Packages

Overview

To install an authorization for the UPDATE package, you require your authorization diskette with the program S5UPG7_0.EXE from the disk packages

- STEP 5/ST V 7.0 - Standard package, UPGRADE or
- STEP 5/ST V 7.0 - Mini PLCs, UPGRADE.

You also require the authorization diskette for STEP 5/ST V6.5 or V6.6 with the counter reading 1 for installation on PCs. There must be no authorization for STEP 5/ST V6.5 or V6.6 on the hard disk.

Procedure

Follow the steps below:

1. Insert your authorization diskette V7.0 in drive A:
2. Type in A:\>S5UPG7_0 and confirm with ***RETURN***.

On PCs follow the steps below:

1. Insert your authorization diskette V7.0 in drive A:.
2. Type in A:\>S5UPG7_0 and confirm with ***RETURN***.

The authorization is uninstalled as described above using the AUTHORS.EXE program.

Warning

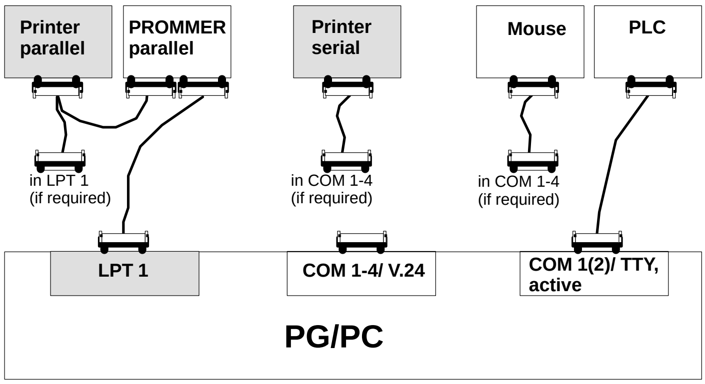
Before initializing or reformatting a hard disk drive, the authorization must be copied back from the hard disk to the original diskette so that it is saved for a new installation. This is only possible with the original authorization diskette. Make sure that you put this diskette away for safe keeping.

2.8 Installing STEP 5 Hardware

2.8.1 Connecting a Printer

Printer Ports

For parallel operation of a printer, use the port LPT 1 (PORT 1, Centronics, Printer) and for serial operation use the ports COM 1 to COM 4.



Which Printers Can Be Used with the Software?

The software supports Siemens printers (known as standard printers) and printers from other manufacturers (non-standard printers). The printer parameters for these printers must be set by loading *DR.INI or using a printer list box. A description of how to do this can be found in the STEP 5 manual.

Note

The devices must only be connected together using the cables when both devices are switched off.

Always secure the cable connectors (screw or lock) whenever possible. This prevents data transfer errors.

2.8.2 Connecting a PLC to the PG

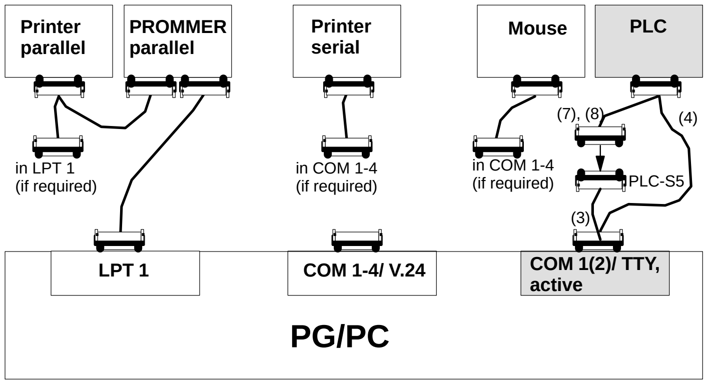
PLC Port

To be able to link up with a PLC, your PG must have an **active** TTY port (20 mA).

If the COM 1 port available is a V.24 interface, the AG-S5 interface must be simulated using an S5 converter

PG with Active TTY COM 1 Port

The programmable controller (PLC) and the PG are connected via a direct connection (4) or by two connecting cables. If the pin assignment described in section 2.8.4 is not used, the connectors will have to be adapted accordingly.



Connecting a PG with an Active TTY Port to a PLC

The PG is switched off.

PG - PLC connection with connecting cable (4) direct or via (3), (7) or (8):

The connectors on the connecting cable (3) with the order no. 6ES5 731-6AG00 are labelled with *PG 7xx COM 1* and *PLC-S5*.

1. Plug the connector labeled *PG 7xx COM 1* into the COM 1 port of the PG.
2. Plug the other end of the connecting cable labeled *PLC-S5* into the matching end of the connecting cable (7) or (8) leading to the PLC.

   It is impossible to mix up the connectors on this cable because they are of different types.
3. Connect the PLC to the remaining free connector.
   Secure the connectors in place.

Connecting Cables for a PG with an Active TTY Port

Connecting cable (3), order no. 6ES5 731-6AG00

Connecting cable (4), order no. 6ES5 734-2xxx0[1)]

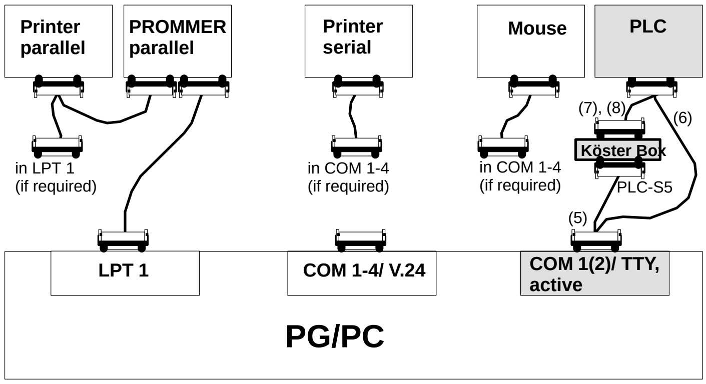
Connecting cable (7), order no. 6ES5 731-0xxx0[1)]

Connecting cable (8), order no. 6ES5 731-1xxx0[1)]

[1)] xxx is the length key. The cables are available in lengths ranging from 1 m to 1000 m. Please refer to catalog ST 59 for details on the length key.

PG with V.24 Port

For a PG with a V.24 port, the port must be converted into an “PLC-S5” port using a V.24/TTY converter (Köster box). The PG is connected to the Köster box directly via a connecting cable with an integrated V.24/TTY converter (6) or via the connecting cable (5). Depending on the type of PLC, the Köster box is connected using connecting cable (7) or (8). These connecting cables must be ordered separately.



Connecting a PG with a V.24 Port to a PLC

The PG is switched off.

PG - PLC connection with connecting cable (6) direct or via (5), (7) or (8):

1. Establish the connection between the COM 1 port of the PG and the Köster box using the connecting cable (5).
2. Plug the connecting cable (7) or (8) into the 25-pin socket on the Köster box and establish the connection to the PLC.
3. Secure the connectors in place.

Connecting Cables for a PG with a V.24 Port

Connecting cable (5), order no. Köster 224 22x[2)]

Connecting cable (6), order no. 6ES5 734-1BD20 (length 3.2m)

Connecting cable (7), order no. 6ES5 731-0xxx0[1)]

Connecting cable (8), order no. 6ES5 731-1xxx0[1)]

1) xxx is the length key. The cables are available in lengths ranging from 1m to 1000 m. Please refer to catalog ST 59 for details on the length key.

2) x stands for the connector type of the PG - Köster box connecting cable

2.8.3 Connecting an EPROM Programmer to the PG

The PGs have an integrated EPROM programming interface. If you are using a PC as a programming device, you can connect an external EPROM programmer. Various devices are available for connection to the parallel or serial port.

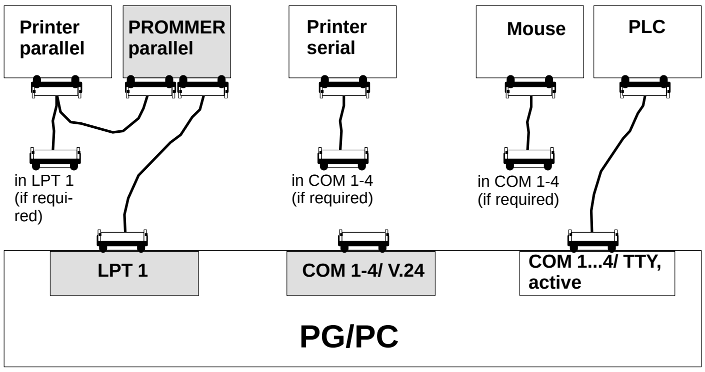
The device connected to the parallel port is known as the **external prommer**.

Parallel Prommer

Port: LPT 1

The cable for the parallel connection is supplied with the external prommer. The external prommer has a connection which extends the parallel port for a parallel printer.

Connection PG-Prommer



Connecting your PG to the Prommer

The PG and the prommer are both switched off.

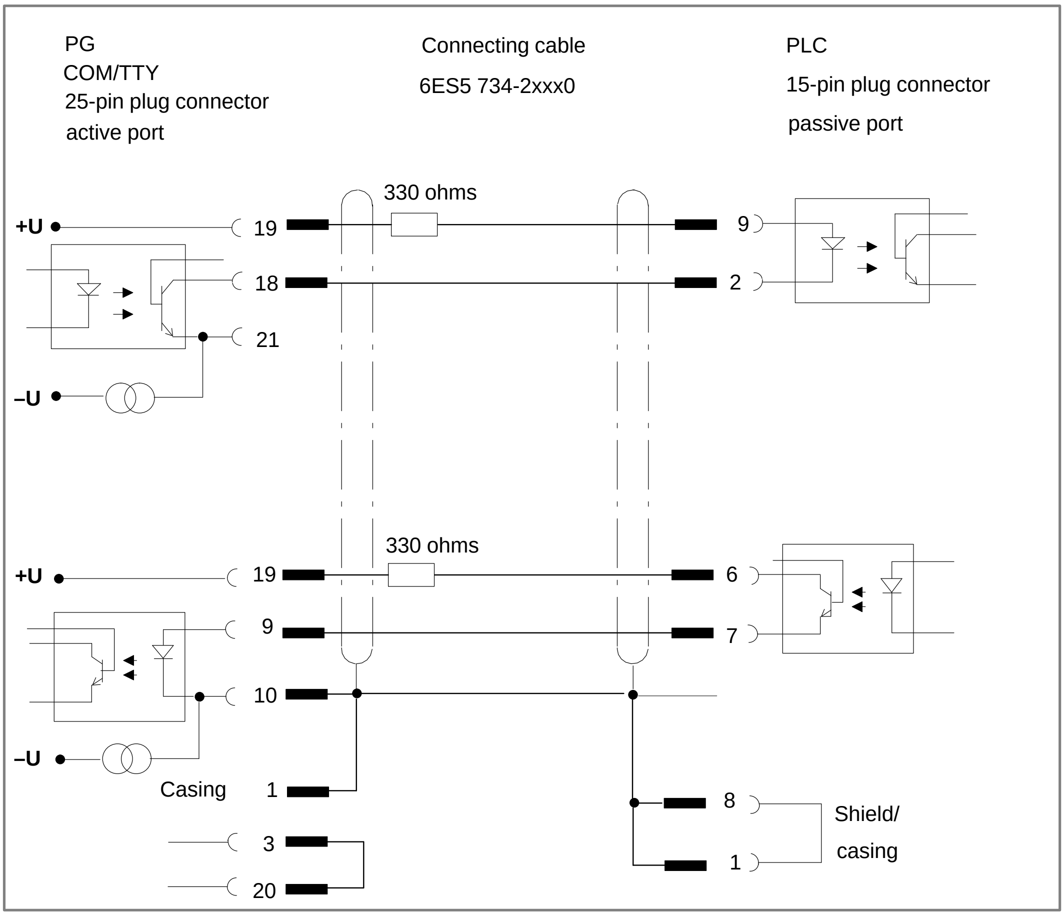
1. Parallel prommer: using the supplied LPT cable, connect the LPT 1 port on the PG with the PC port on the external prommer and, if applicable, connect your parallel printer to the Printer port of the external prommer.
2. Any connectors fitted with screws or clips must be secured.

2.8.4 Overview – Connecting Cables to PLC, Partner PG, Prommer

Connecting cable no.	Order number	Connection from (Connector on PG)	to
3	6ES5 731-6AG00	PC COM 1 (PG 7xx: 25-pin male)	Connecting cable 7 or 8 (PLC) Connecting cable 10 (partner PG)
4	6ES5 734-2xxx0 1)	PC COM 1, 2 25-pin female	PLC 15-pin female
5	Köster 224 22x	PC COM 1, 2	Köster box
6	6ES5 734-1BD20	PC COM 1, 2 25-pin female	PLC 15-pin female
7	6ES5 731-0xxx0 1)	Connecting cable 3 or Köster box	PLC 25-pin male
8	6ES5 731-1xxx0 1)	Connecting cable 3 or Köster box	PLC 15-pin female
10	6ES5 733-2xxx0 1)	Connecting cable 3 or Köster box	Partner-PG COM 1

1) xxx is the length key. The cables are available in lengths ranging from 1m to 1000 m. Please refer to catalog ST 59 for details on the length key. A maximum cable length of 3 m is permitted for use with a prommer.

Connector Assignment of the Active TTY Port



2.8.5 Installing STEP 5 Drivers

Selecting and Deselecting Drivers for STEP 5/ST V 7.0

You can select and deselect MS-DOS drivers for SINEC L2 or H1 (SIMATIC NET network drivers) for STEP 5/ST with the S5DRV.EXE program.

The drivers are activated or deactivated by an entry in the AUTOEXEC.BAT file. The original file is saved as AUTOEXEC.S5 prior to the modifications. The changes are only effective after rebooting the PC.

To call the program:

Table 2-3 Calling S5DRV

Operating System	Call
MS-DOS	Type in the command S5DRV.
Windows3.x	Start the program by double-clicking the STEP 5 drivers icon in the STEP 5 program group.
Windows95	Click S5 driver installation in the start menu / STEP5V70.

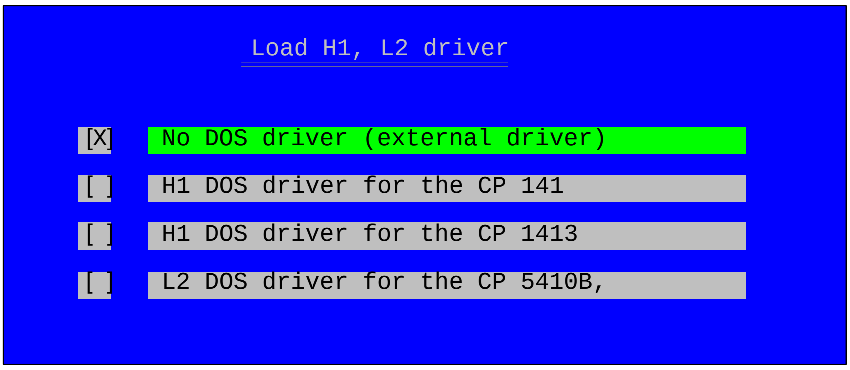


Figure 2-16 Installation Options

The S5DRV.EXE program is menu-guided. You can control the program using a mouse, a trackball, or the cursor keys and function keys.

In the menu, you can select the MSDOS drivers for SINEC L2 or H1 (SIMATIC NET network drivers) and SIMATIC NET network functions.

Defaults

When shipped, no STEP 5/ST software drivers are activated. The current status of the selection is displayed as follows:
[X] =
[ ] = not selected.

Note

You should select drivers for STEP 5/ST to suit your specific requirements so that you use as little memory as possible making more memory available for STEP 5 itself.

You can only select MSDOS drivers for SINEC L2 or H1 (SIMATIC NET network drivers) when you have already installed the corresponding driver software on your PC.

2.8.6 STEP 5 Keyboard Editor

You will only need the keyboard editor if you want to use a personal computer (PC) as a programming device (PG).

Using the keyboard editor, you can adapt the key assignment to match STEP 5. You can add key combinations or modify existing combinations.

With the keyboard editor, you can also switch over between a color monitor and a monochrome monitor.

If your PC has a keyboard other than the international standard keyboard or if you wish to assign additional key combinations for STEP 5, you may want to change the key assignments. If your PC has a standard keyboard (MFII keyboard), you will be able to work comfortably with the existing key assignments.

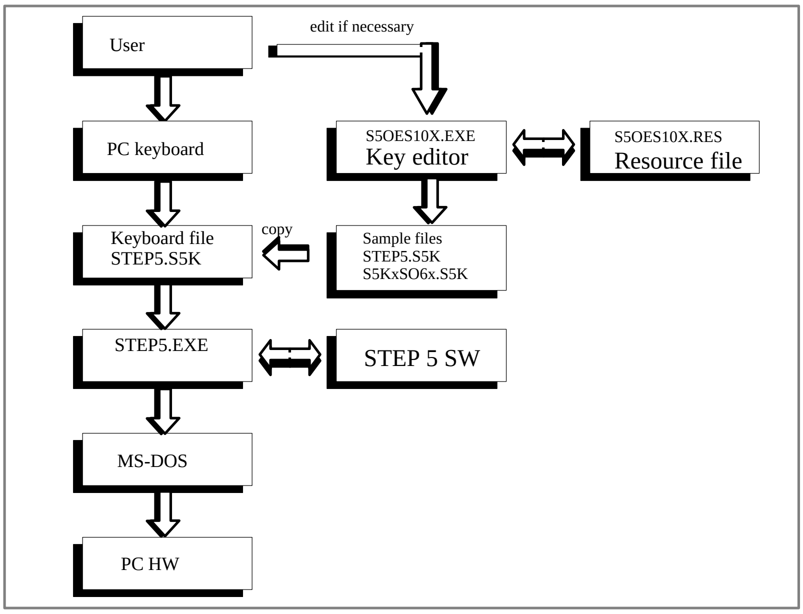
In the default setting of the editor the keyboard assignments are given in German; you can, however, select another language.

Files

The files are found in the directory DR:\STEP5\S5_SYS.

File	Description	Content
S5OES10X.EXE	Keyboard editor	Editor; manages and documents user interface
S5OES10X.RES	Resource file	Selects preset keytop texts (the respective keyboard driver must be loaded)
S5KXS06X.S5K STEP5.S5K	Keyboard file	File with new keyboard assignment must be copied to the home directory where they can be edited. STEP5.S5K = keyboard file for STEP 5/ST V7.0 parts S5KXS06X.S5K= keyboard file for STEP 5/ST V6.6 parts
S5KXS01X.S5K or S5KXS01K.S5K	Keyboard file	Keyboard assignment (in the directory \STEP5; if this file is not present, STEP 5 loads the default setting)

Data Flow



Requirements

If you want to load another keyboard assignment for STEP 5, you should be familiar with the following:

- the S5 keyboard itself,
- the functions of the S5 keyboard, and
- the file structure of the S5 software.

You will find an explanation of the S5 keys in the Appendix.

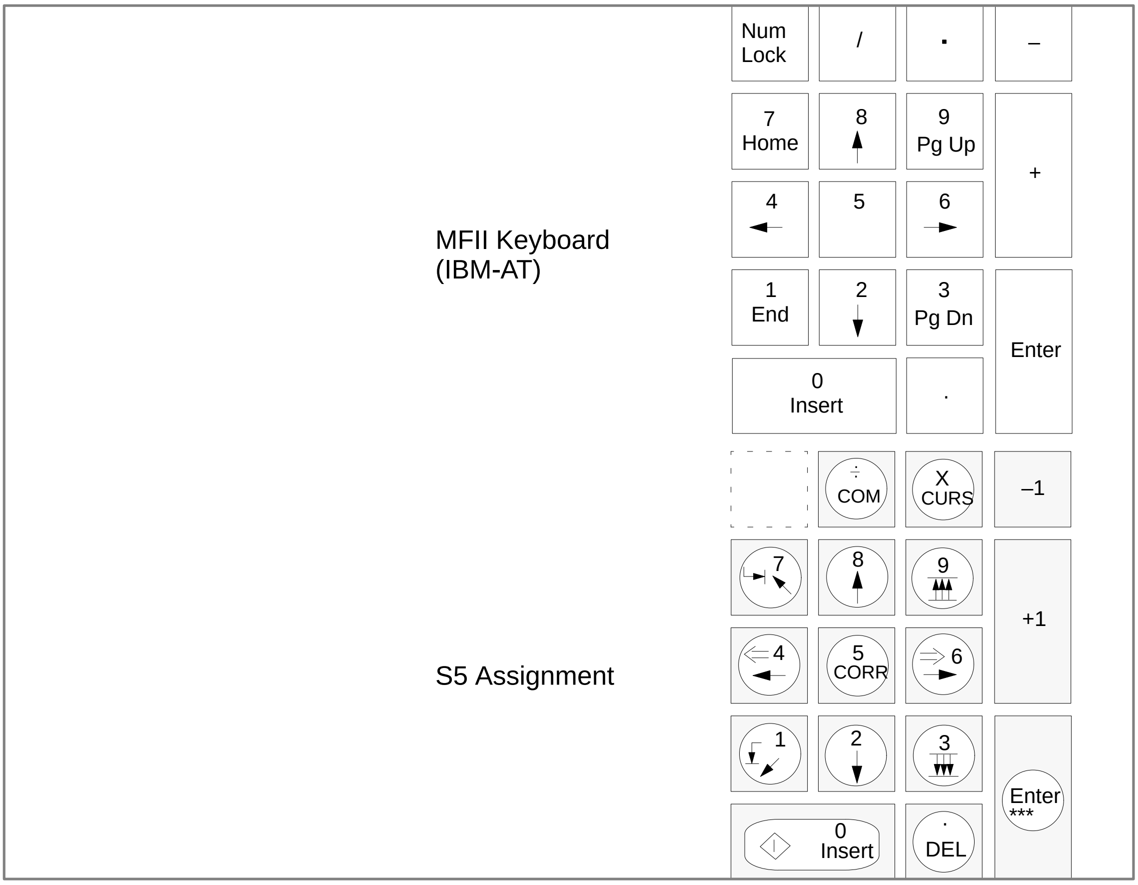
Keyboard with S5 Default Assignment

Notes

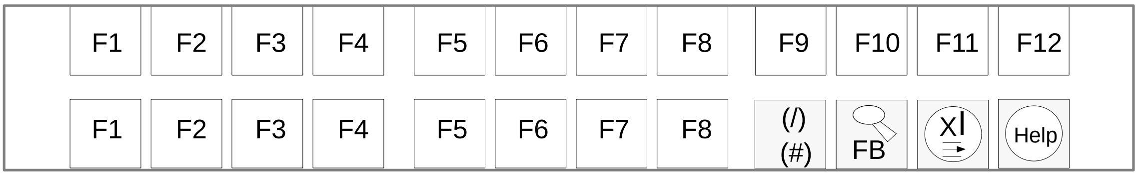
Please note the following limitations:

On the PG 750/770 the keys < COM > (Comment) and <***> (Shut down network) have no function.

To select these functions use the keys **SHIFT F6** <NW–Comm.> (Change to network title and comment) or **F6** <Shut down network> (Shut down current network and start a new one).



Function Keys

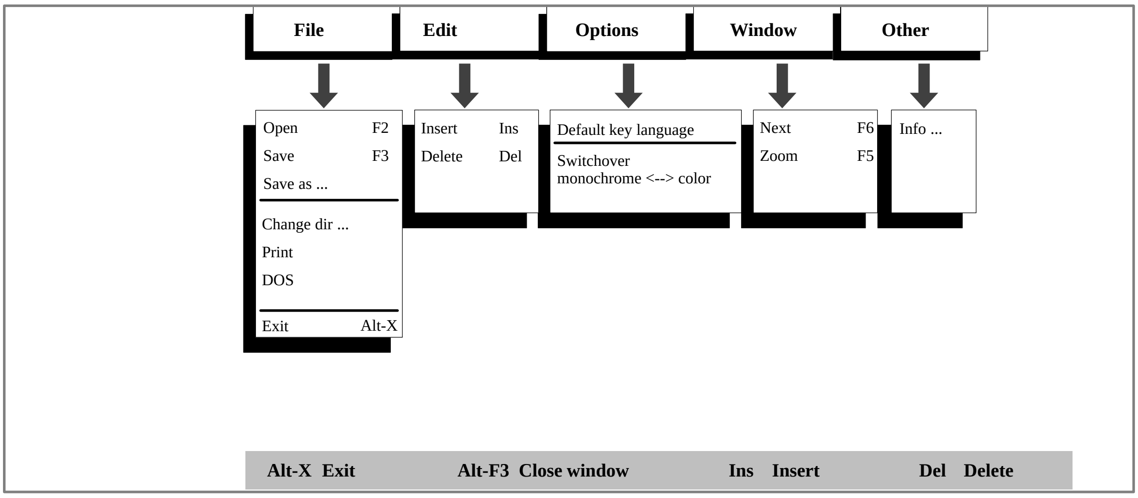


Calling the Keyboard Editor

Table 2-4 Calling the Keyboard Editor

Operating System	Call
MS-DOS	Type in the command S5KEDIT.BAT
Windows 3.x	To start the program double-click on the icon S5KEDIT in the STEP 5 program group.
Windows95	Click the Start menu/STEP5V70/S5 keyboard editor.

Keyboard Editor Menu



User Interface

Menu	Menu options	Function	Display
File	*Open* ***F2***	Opens the keyboard for editing	Selection box *Open file*
	Save ***F3***	Saves the current keyboard file	Current window
	Save as ...	Saves the current file under another name in the same directory or in another directory	Selection box *Save file as*
	Change directory	Opens another directory	Selection box *Directory...*
	Print	Prints current file	Unchanged
	DOS	Changes back to DOS command level (return using ***EXIT*** and ***Return***)	DOS command line
	Exit ***ALT*** *+* ***X***	Exits the editor	DOS level
Edit	*Insert* ***Ins***	Inserts or changes keytop texts at the cursor position	A further window *Assign S5 function to another key combination*
	Delete ***Del***	Deletes keytop text at cursor position	Window with deleted assignment
Options	Default key language	Selects a resource file in another language (S5OnS10X.RES)	Selection .RES *Open file*
	Switchover monochrome/ color	Switches the screen from monochrome display to color display	Screen in black/white or color
Window	*Next* ***F6***	Updates the bottom window when several are open at once	Current window
	Zoom	Increases size of current window	Current window
Other	*Info ...*	Information on the release	Information screen

Default Keyboard Language

You will need to use this language selection for a new file and if you require a keyboard language other than German.

Otherwise, German is the default keyboard language for any new file.

Language Selection

1. Select **Default key language** in the **Options** menu.

   The dialog box *Open file* (file type *.RES) appears with a selection of the existing resource files.

2. Select the resource file for the required language or enter a new name in the following syntax:

   S5O x S10X.RES

   x = D German

   E English

   F French

   I Italian

   S Spanish

   Click on the **Open** button.

   The keyboard language is set.

Open File

1. Select the menu command **File** ▸ **Open** (***F2***).

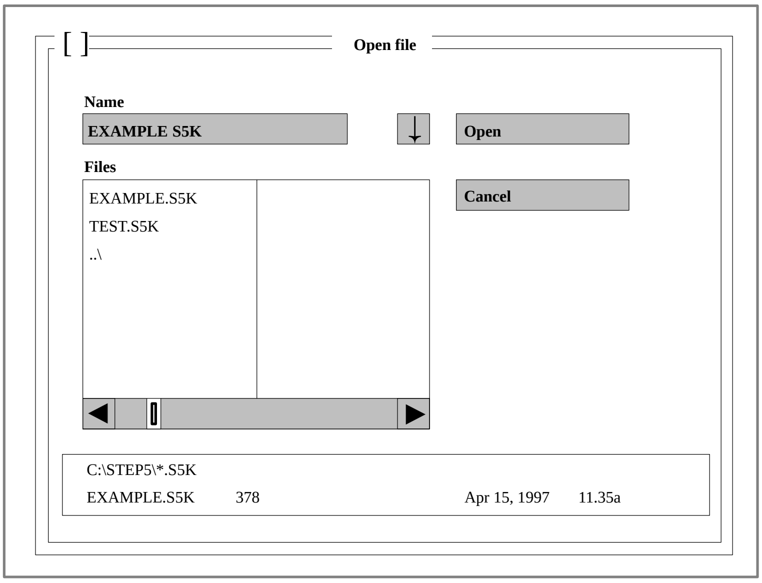
   A list of files of the type *.S5K is displayed.

2. Select a file or enter a file name.
3. Click on the **Open** button.

   The file is opened for editing. For a new file the default assignment is displayed.

4. Open any additional file(s) as in steps 1 to 3.

“Open File” Dialog Box



Inserting a Key Assignment

Requirement: you have opened a key assignment file.

1. Select the required key in the current file using the mouse or cursor keys.

   The selected line is highlighted on a colored or gray background, the line number is shown in the lower left corner of the window.

2. Select **Edit ▸ Insert** or press the ***Insert*** key.

   In the lower third of the screen an additional window appears *Assign S5 function to another key combination*. The description of this function is shown in the first line of this window.

   Note: After using the ***PRINT*** and ***PAUSE*** keys, the input must be enabled again via the keyboard with the key combination ***CTRL + SHIFT***.

3. Select the new key combination by pressing the corresponding keys.

   The *Save assignment* window appears or the message *The key (combination) is already assigned internally*.

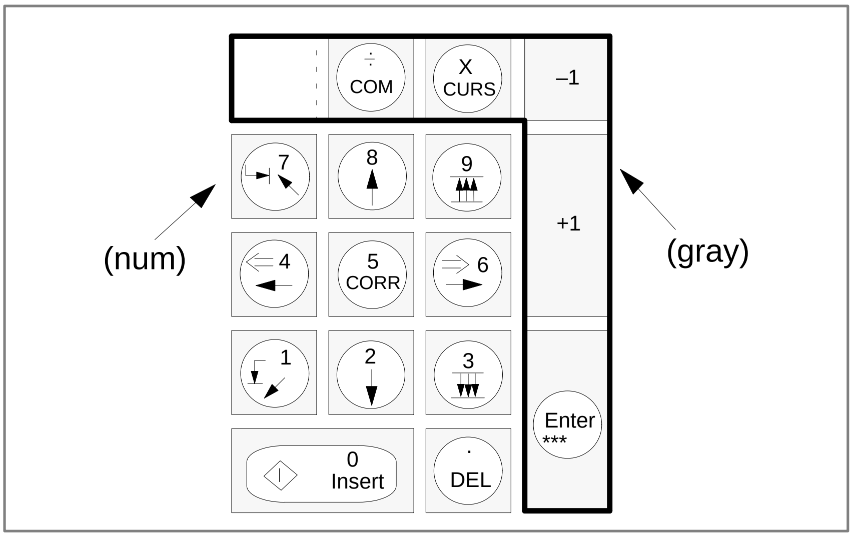
4. Assign the new key combination by **overwriting** or **inserting** the new key combination.

Restrictions

For some key combinations the keytop text displayed does not correspond to the printed alphanumeric key.

- The ***D*** key on the PG keyboard cannot be assigned any function.
- Hotkeys which are preassigned by resident programs at the time when the key editor is being used cannot be assigned.

Key Descriptions in the Editor



Default Assignments

- Keytops which are only present on a 101/102 keyboard are marked with (101).
- Keytops which are only present on a PG keyboard are marked with (PG).
- S5 functions which are only for GRAPH 5 are marked with (GRAPH 5).

Deleting a Key Assignment

Requirement: You have opened a key assignment file.

1. Select the key assignment you want to delete.

   The selected line is highlighted on a colored or gray background, the line number is shown in the lower left corner of the window.

2. Select **Edit ▸ Delete** or press the ***Del*** key.

   The key assignment in the selected line is deleted from the screen.

   Note: After using the ***PRINT*** and ***PAUSE*** keys, the input must be enabled again via the keyboard with the key combination ***CTRL + SHIFT***.

3. If necessary, undelete the assignment by clicking on the square box in the top left of the window frame and clicking on *No*.

   With *No*, all the changes made since you last saved are lost.

Saving Key Assignments

- Select **File ▸ Save** or
- Select **File ▸ Save as** to save the file under another name or in another directory.

Changing the Working Directory

1. Select the menu command **File ▸ Change dir...**
2. Browse through the displayed tree structure. If you double-click on a directory, its subdirectories are displayed.
3. Click on the required directory and the ***Chdir*** button.
4. Save the setting by clicking on the ***OK*** button.

Activating the Key Assignment

Using the menu function **File ▸ Save as** you can copy your file (assignment) to a new file. When you want to activate your new key assignment for STEP 5, you must copy it into the STEP5\ST home directory under the name STEP5.S5K (STEP 5/ST V7 parts) or under the name S5KxS06x.S5K (STEP 5/ST V6.6 parts).

Copying the Assignment

1. Open the key assignment with the menu command **File ▸ Open** and open the file of the type *.S5K.
2. Select **File ▸ Save as**.
3. Open the directory STEP5\ST home directory.
4. In place of the asterisk, enter the name of the keyboard file **STEP 5 or S5KXS06K**.
5. Click the ***OK*** button or press ***ALT*** + ***W***.

Printing the Assignment

1. Open the key assignment with the menu command **File ▸ Open** and open the file of the type *.S5K.
2. Select **File ▸ Print**.

Switching to the DOS Level

You can only call the DOS command level when there is sufficient free user memory.

1. Select **File ▸ DOS**.

   The command line of the DOS level is shown.
2. Type in a DOS command.
3. To return to the editor, type in the ***EXIT*** command.

Switching Between a Color and Monochrome Screen

In the keyboard editor, select the menu items **Options ▸ Switchover monochrome ↔ color**.

Testing the Resource File

1. Open a new file of the type *.S5K with the keyboard editor menu **File** ▸ **Open** ▸ **Enter file name**.
2. Save without any changes with **File** ▸ **Save**.
3. Close the window with ***ALT*** *+* ***F3***.
4. Open the same file again.

Result: If no error occurs when the file is read, the assignment is correct.

If an error occurs, check whether any key combinations are occupied by two or more assignments.

Information

Information about the release of the software can be obtained by selecting *Other* and then *Info...* in the menu.

You quit the screen by clicking ***OK***.

Returning to the DOS Level

You can return to DOS by pressing the keys ***ALT +*** *X* or by selecting the menu command **File** ▸ **Exit**.

2.9 Working with COM Packages

When working with COM packages, remember the following points:

- When using COM packages, only one DOS directory per drive can be used.
- No drives with a driver letter higher than P: must be used since the COM packages have not been upgraded to the V7.0 level.
- With COM packages, remember that the system directory of STEP6 V7.0 is different from the system directory of the COM adapter. The COM packages use their own system directory ...\S5_SYS\S5_COM.
  This division is necessary to allow the COM packages to run.
- COM packages can be included in the Change menu so that they can be started directly.
- COM 155H and COM 95F can be operated under STEP 5/ST V 7.0 in the *Change* ▶ *Others* menu as optional packages. Their previous link to the user interface of Version 6.x can no longer be used in Version 7.0.
- Various COM packages require the default files from the project settings (??????PX.INI).
  The set file `DR:\<Directory>\<Filename>` in the tab page is therefore only valid for the STEP 5 session.
- COM packages use some of the names of the file(s) set in your defaults, but they cannot access them. To be able to use the set files in COM packages, these must be copied to the directory of the COM package.
  This can involve the following files that are required by various COM packages in their own directory:

Name	File Name
Program file	??????ST.S5D
Symbol file	??????Z0.INI
Footer file	??????F1.INI (80 characters) ??????F2.INI (132 characters)
Printer file	??????DR.INI
Output file	??????LS.INI
Path file	??????AP.INI (+ path name)

2.10 Compatibility

STEP 5/ST V7.0 is compatible in terms of software with Version 6.6. Using the menu item **Change ▸ Others..**. you can load parts of Version 6.6.

In terms of compatibility, note the following points:

- The PG 710 I/II is no longer supported (for STEP 5 V7.0, a minimum of 4 Mbytes of memory is required. These PGs cannot be upgraded.)
- Serial prommer no longer supported.
- Existing key macros must be recreated.
- The alternative BTRIEVE data management is no longer supported.
- Support for diagnostic/setpoint data based on the CP 551 is no longer available.
- GRAPH 5/II V6.x cannot be operated under STEP 5/ST “V7.0”.
- Older project files (PJ.INI) can be converted to the V7.0 level using integrated conversion tools to allow the features above to be used. By keeping to compatibility criteria (no drives higher than J: or P:, only one directory per drive) it is possible to reconvert project files to the V6.x level.
- To distinguish them, the new project files end with PX.INI.
- Under certain circumstances, minor adaptations of existing user files for Version 6.x maybe necessary to allow you to use the extended options of the DOS file system. This applies not only to the project files (PJ.INI) but also to bus path files (AP.INI) and DOC command files (SU.INI).
- If you make use of the new options of working with several DOS directories, you will receive a message indicating that compatibility with older STEP 5 versions will be lost.

User Interface

3

Overview

The STEP 5 software was developed according to modern ergonomic principles and is therefore to a large extent self-explanatory.

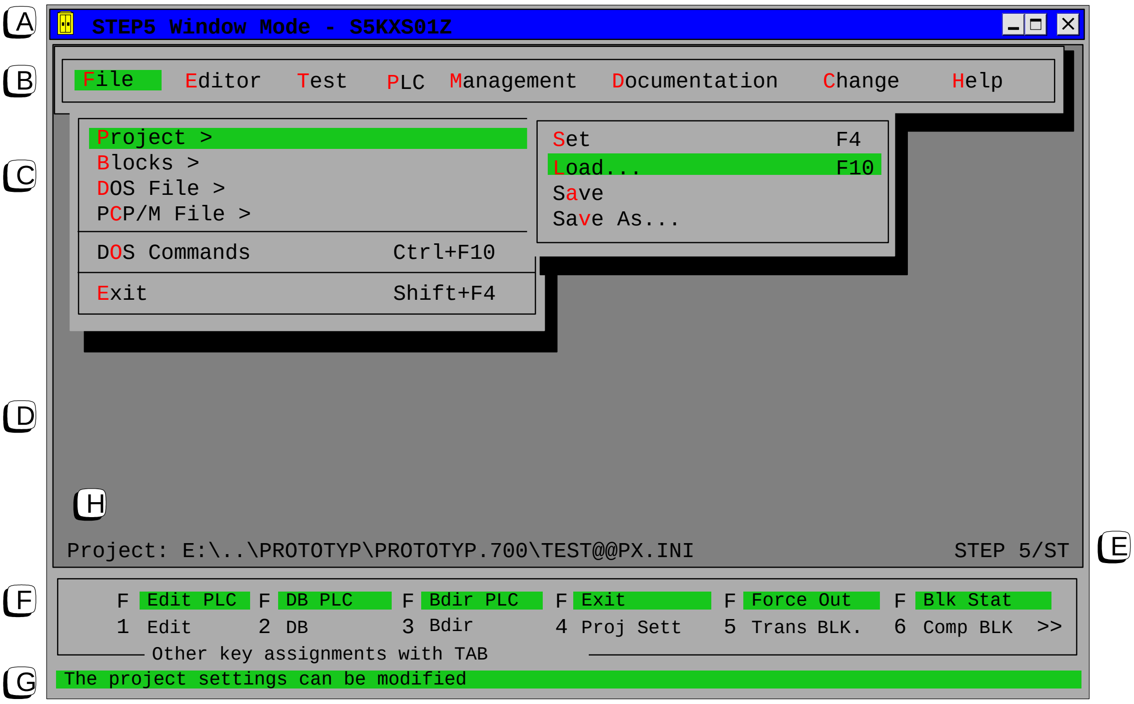
If you have not yet worked with this type of user interface, reading this chapter will familiarize you with the most important input elements and the terminology.

Chapter Overview

3.1 Selecting Functions in the Main Menu

Overview

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.



(A) Title Bar

The title bar has the name STEP 5. The buttons shown in the title bar are those familiar from Windows95. The title bar is not displayed in the full screen mode or under MS-DOS.

(B) Menu Bar, (C) 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.

If you select menu items with dots (...) to the right of them, you open a dialog box.

(D) Working Area

The dialog 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.

(E) S5 Identifier

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

(F) Function Key Menu

The function key menu allows you to call certain list boxes or editors directly without a longer series of keystrokes.

To display the remaining 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 (see also Section 3.2):

- 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 ***F12***).
- 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 F12***).
- In some situations, a combination of the function keys with the Ctrl/SHIFT + Ctrl key is also possible.

Help

You can obtain more detailed information about the functions assigned to the keyboard by activating the **Key Assignment List** function in the **Help** menu with ***Ctrl+F12***.

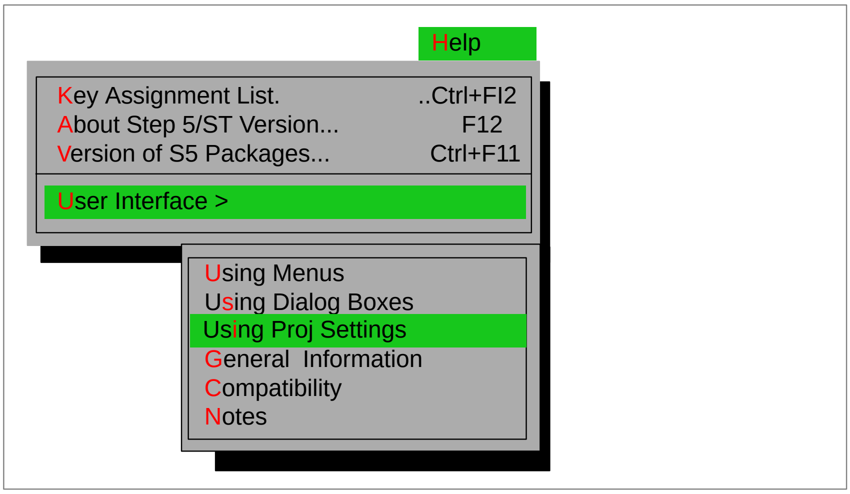


Figure 3-1 Help Menu

(G) Info Line

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

(H) Project File

This information line displays the project file (PX.INI) you are currently working with.

3.2 Input Elements

User Interface

The user interface of STEP 5/ST was designed so that all functions can be activated with the keyboard or with a mouse.

Hotkeys

To allow more convenient operation with the keyboard, the display includes hotkeys. These hotkeys are letters and numbers highlighted in a color and by pressing the corresponding key you can select and activate functions more quickly. After you press a hotkey, wherever the cursor is currently positioned, the software jumps to the corresponding point on the screen or triggers the required function in a menu.

Upper Menu Bar

Using the key combination ***ALT+ Letter***, you open the required submenu from any point in the program.

ALT+F : Opens the File menu or

ALT+T : Opens the Test menu

Submenus

In these menus you activate or select a function simply by pressing the colored hotkey. Only the hotkeys of the currently opened menu are active.

Dialog Boxes

Within dialog boxes you can use the hotkeys to navigate to different positions in the boxes. In dialog boxes, you once again use the combination of the ***ALT*** key with the hotkey.

Key Macros

Within STEP 5, it is possible to record key strokes, for example within the block editor.

This allows you to automate various steps.

You select the key macro program as follows:

CTRL+ALT+D

The *Select macro* dialog box is displayed in which you can make the following settings:

- Type in (or select) the macro file (......TX.INI)
- Type in a title
- Run a recorded key macro
- Record a key macro of your input
- Run a recorded key macro in the single step mode

If the option `[ ] Run in single-step mode` is set, you must press the key combination ***CTRL+ALT+T*** when running the recorded macro to activate each single step.

You complete a recording with ***CTRL+ALT+D***

You can cancel the key macro mode with the ***ESC key.***

Note

It is not possible to operate STEP 5/St with the mouse or cursor during the recording of a key macro.

When using the hotkeys, you should note that the key assignment differs from language to language.

The START@TX.INI macro automatically starts the running of a recorded macro when you start STEP 5/ST.

Key macros cannot be continued after you use the command **File ▸ DOS-Commands** or **Change ▸ Others.**

Make sure that you note down the start and end of a recording.

Keys in the Function Key Menu

Some of the submenus are nested when they are open. You can close a submenu with the ***ESC*** key without triggering a function. If you press the ***Return*** key, you trigger a function or open a submenu.

To keep the selection of commonly used functions as simple as possible, function keys (***F1*** to ***F12***) were defined for most submenus and these are effective at any point in the menu.

The ***F1*** key, for example, calls the job box for the block editor, ***Shift+F3*** displays the block directory on the PLC.

The assignment of the function keys is displayed at the lower edge of the screen when you are at the menu level. Since this assignment also includes the combination of function keys with ***Shift*** or ***Ctrl***, you can display the next function key assignment level using the ***TAB*** key.

If you select **Help ▸ Key Assignment List**, you can display an overview of the function keys used.

The assignment of the function key bar is always visible when the menu is active. Optional packages may have their own assignment for the function keys.

With the ***TAB*** key, you can move from one function key assignment level to the next. If you prefer to use the mouse, click the **>>** at the bottom right of the function key bar.

3.3 Selecting Functions

Calling Functions

You call a function or an editor in two steps:

1. Select the function in a main or submenu
2. Complete the input fields in the job box and confirm your input.

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

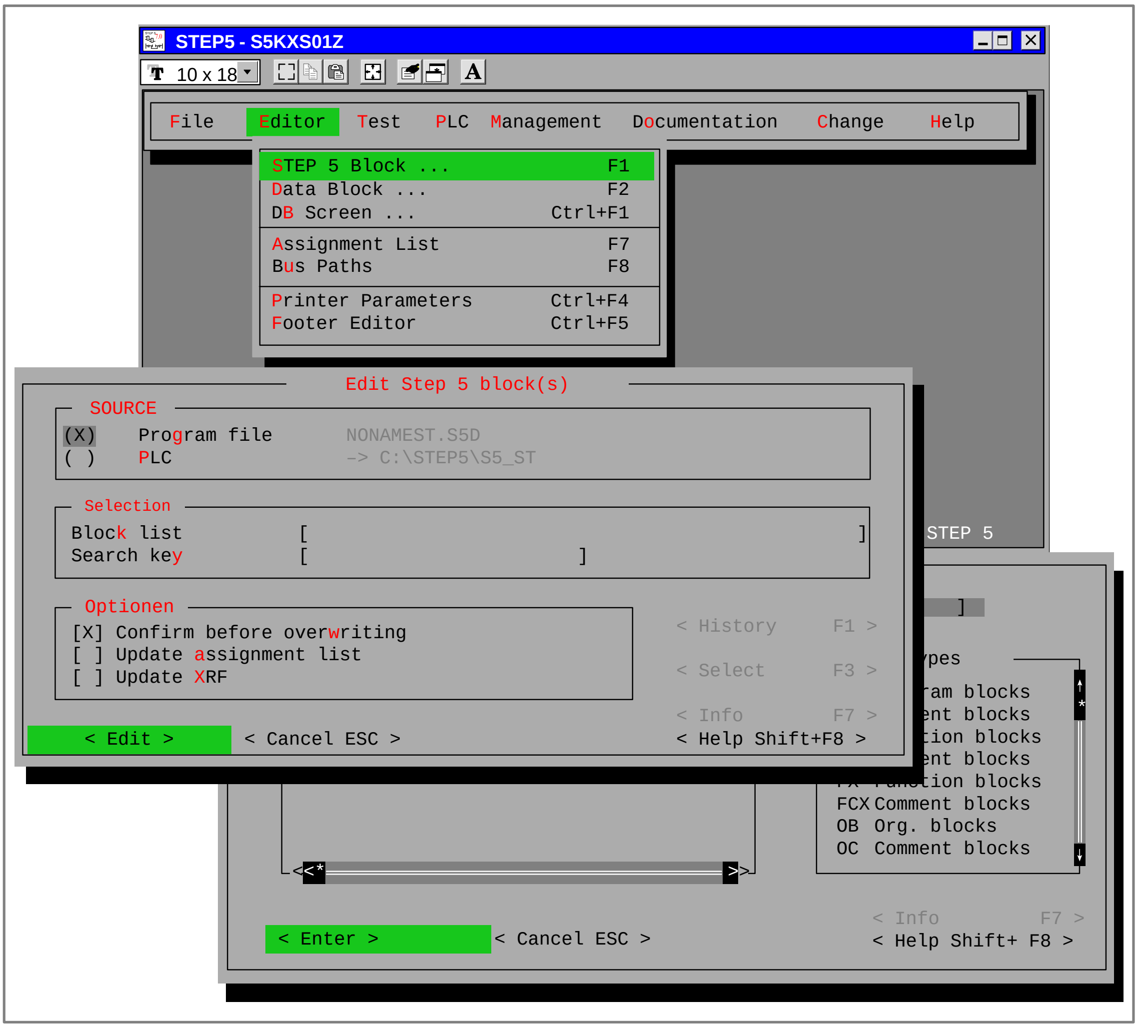


Figure 3-2 Selecting Functions in Main and Submenus

3.4 Using Help Functions

Online Help

The online help system provides information at the point at which you require it. You can find specific information quickly without having to refer to the manuals. The online help includes the following:

- **Help topics**: Provides various ways of displaying help information, see Figure 3-3.
- **Context-sensitive Help:** The button ***<INFO F7>*** or **F7** key: these display information about the selected object or the active dialog box or window.
- **Using Help**: See Figure 3-3 menu **Help** ▸ **User Interface** ▸ or the ***Help Shift+F8*** key displays a description of the options available to find certain information in the help system.
- **About**: Displays information about the current version of the application.

Calling Online Help

You can call the online help system in various ways:

1. In a dialog box, click the button ***Help Shift+F8*** or press the ***SHIFT+F8*** key. You then obtain general help about this dialog box. You can scroll and page through these multi-page texts.
2. Position the mouse pointer in a window or dialog box on the topic about which you require help and click the ***Info F7*** button or press the ***F7*** key.
3. Select a menu command from the *Help* menu in the menu bar.

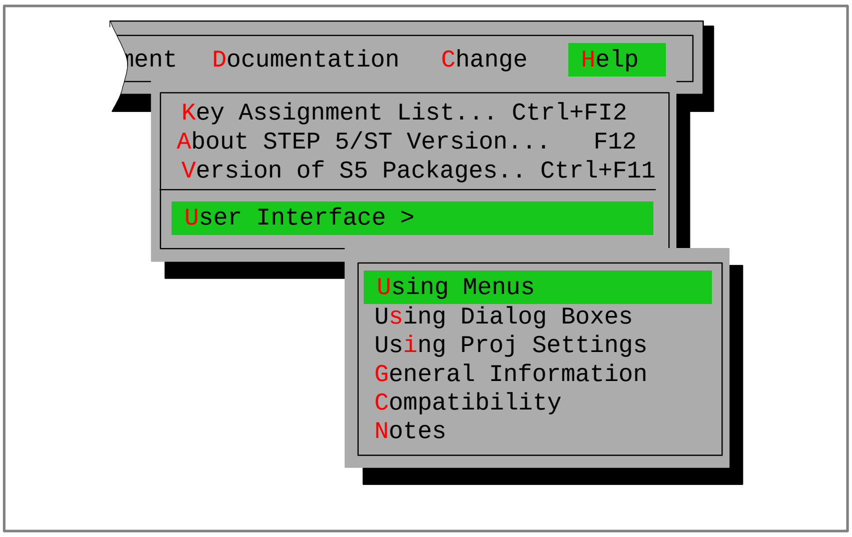


Figure 3-3 Help Menu

3.5 User Interface: Dialog Boxes

Making Entries in Dialog Boxes

In dialog boxes, you can enter information that is required to execute a particular task. There are four types of dialog box available:

- Job box (see Section 3.6)
- Tabs and tab pages (see Section 3.7)
- File/directory selection (see Section 3.8)
- Block selection (see Section 3.9)

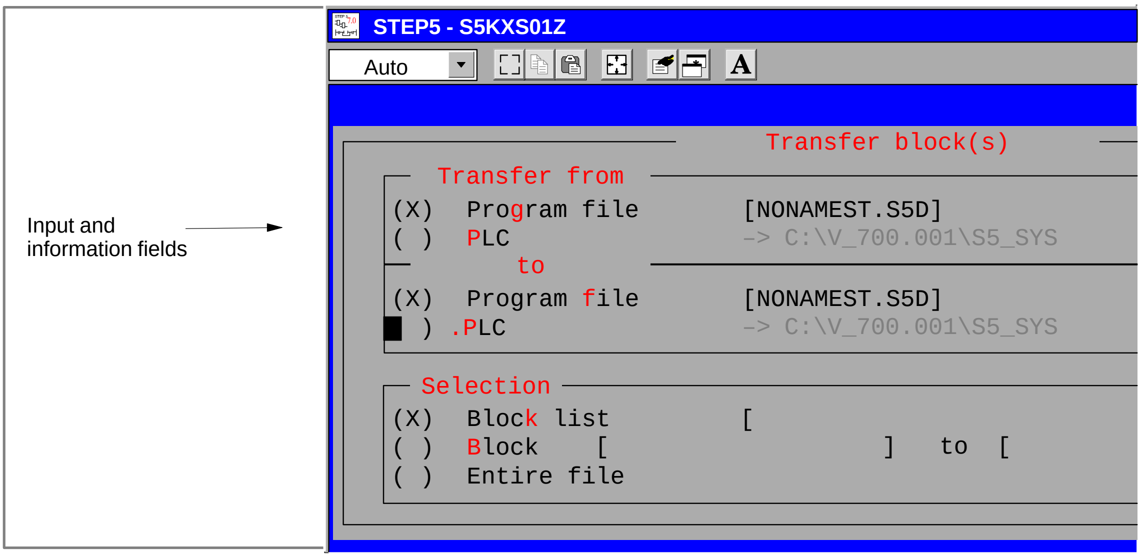


Figure 3-4 Example of a Dialog Box (here a Job Box)

3.6 Job Box

Function

The job box is a dialog box for calling an editor or a function. The information used in these forms can have effects on the elements with the same name in the project settings. The most important components of dialog boxes are explained based on an example in Figure 3-5.

Input Elements

As far as possible working with dialog boxes has been made uniform and is based on the strategies used in Windows programs.

Dialog Elements

Table 3-1 The Dialog Elements

Element	Function
( ) Option button	You can select one option from several alternatives using the cursor keys or the mouse.
[ ] Check box	You can select one or more optional settings with the ***F3*** key, spacebar or mouse.
Select box:	If you press the ***F3*** key, a list box appears in which you can select predefined settings. If there are only two options available, you can toggle between them with the ***F3*** key.
List box	You select an element in the list using the ***Return*** key or by **double-clicking** with the mouse (see also Section 3.8 or 3.9)
Text box [...]	In text boxes, you type in your input using the keyboard, for example file names. In these boxes, alphanumeric characters are permitted (but no umlauts).
< History F1 >	You can select one of the last 20 entries.
< *Edit* F2 >	Calls the editor for the specified file.
< Select F3 >	Activates a selection in a dialog box or list box or by marking an element.
< Info F7 >	Information about completing the text boxes.
< Help Shift+F8 >	General information about dialog boxes.
< Cancel ESC>	Input is canceled.

You can also achieve the same effect as clicking a <...> button by pressing the corresponding function key (see Table 3-2).

Table 3-2 Function Keys Corresponding to the < > Buttons

Function Keys

Function Keys	Effect
F1 *= History*	Selects one of the last 20 entries.
F2 *= Edit*	Calls the editor for the specified file.
F3 *= Select*	Activates selection via a dialog box or list box or by selecting an element.
F4	
F5	
F6	
F7 *= Info*	Information about completing text boxes.
Shift+F8 *= Help*	General help on dialog boxes.

Different function keys are permitted depending on the position of the green cursor bar. Disabled function keys are displayed in gray.

Table 3-3 Special Keys for Text Boxes

Keys with Special Functions

Keys	Effect
Num. 5:	Switchover between the insert and overwrite mode.
SHIFT+Del:	Text box is cleared.

Applies only to text boxes!

Table 3-4 Working with the Mouse and Keyboard

Mouse, Keyboard

Keys	Effect
Cursor keys	Changes between option boxes
TAB key	Changes between the input elements of a dialog box, ***TAB*** moves forwards right/down, ***Shift+TAB*** backwards left/up
Mouse	Positioning with a single mouse click, double-click has the same effect as the return key
Hotkeys	Direct selection of an element with ***ALT+***letter or ***ALT+***number.
Return key	You trigger a function with the ***RETURN*** or ***Insert*** key if the entries in the dialog box are correct. You activate a search dialog in text boxes with (?) and (*) You activate the drive/directory in the Dr/directory list boxes

“Memory” of the Job Boxes

The STEP5 job boxes have a so-called “memory” that buffers the last specified contents on hard disk so that the contents of the dialog boxes are recorded for the present STEP 5 session or for more than one session and are ready the next time the box is called. The memory includes the following:

- Status of text boxes and other dialog elements
- History of text boxes
- Output to screen or not to screen.

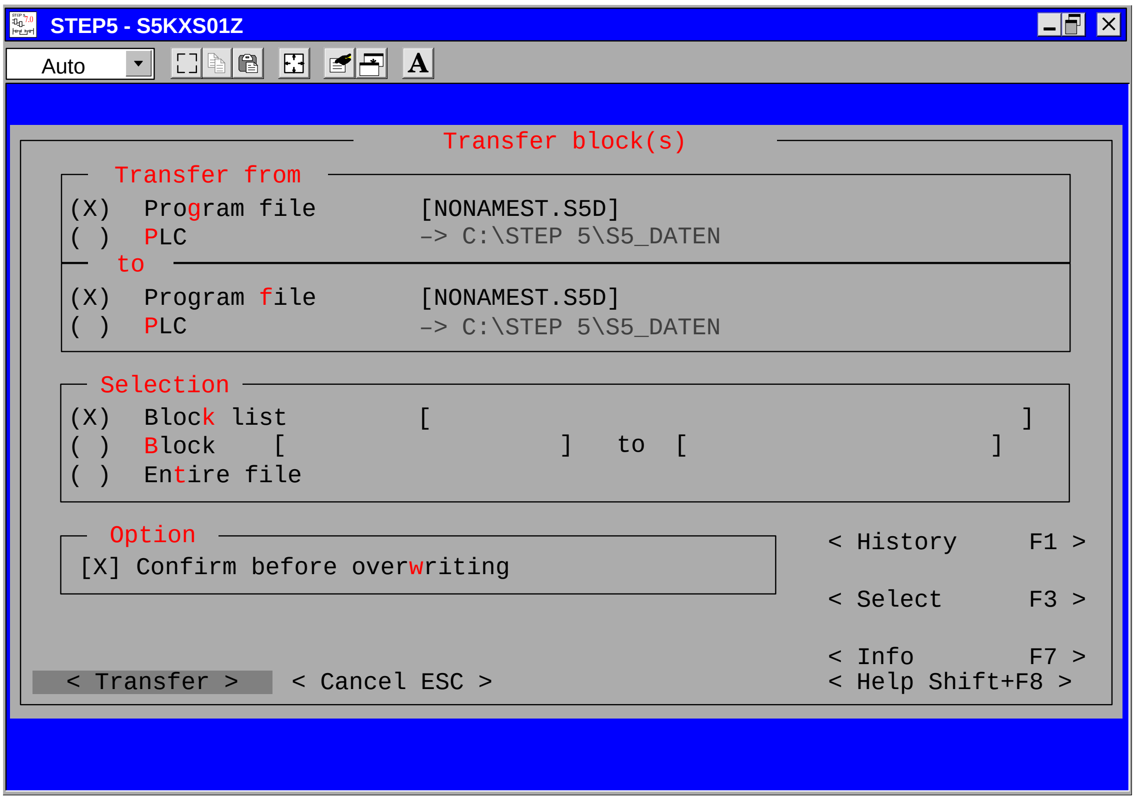


Figure 3-5 Example of a Job Box

3.7 Tabs and Tab Pages

Function

The content of the dialog box that you obtain after activating the menu command **File ▸ Project ▸ Set F4** is organized in tabs to make it easier to work with. Each tab is clearly named and you simply click the name tab to bring a particular tab page to the foreground.

3.7.1 Working with Tabs

The elements available for working with tabs are basically the same as in dialog boxes.

The dialog consists of several tab pages with the current tab page covering the others.

Using the hotkeys ***ALT +*** **number**, you can change between the tab pages. You can also change to a different tab page by clicking the tab with the mouse.

Function Keys in Tab Pages

Keys	Effect
F3	The cursor must be located on the name of a text field. 1. Parameter settings can be changed with ***F3*** (for example YES/NO or RW/PROT). 2. A list box is activated. Select entries with the cursor. You accept an entry from the list by pressing the ***Return*** key or by double-clicking. 3. A file list box is displayed.. You can navigate through the box and select a file (*job box*).
ALT + Tab Number	Changes to the tab page, for example ***ALT + 2*** brings tab page 2 to the foreground.
F7	An information text is displayed for the field marked by the cursor.
Shift F8	Displays general help information.

Memory

When you exit the tab dialog, STEP 5 records the currently active page and the cursor position in this page and selects this the next time you open the project settings. This applies only within a STEP 5 session.

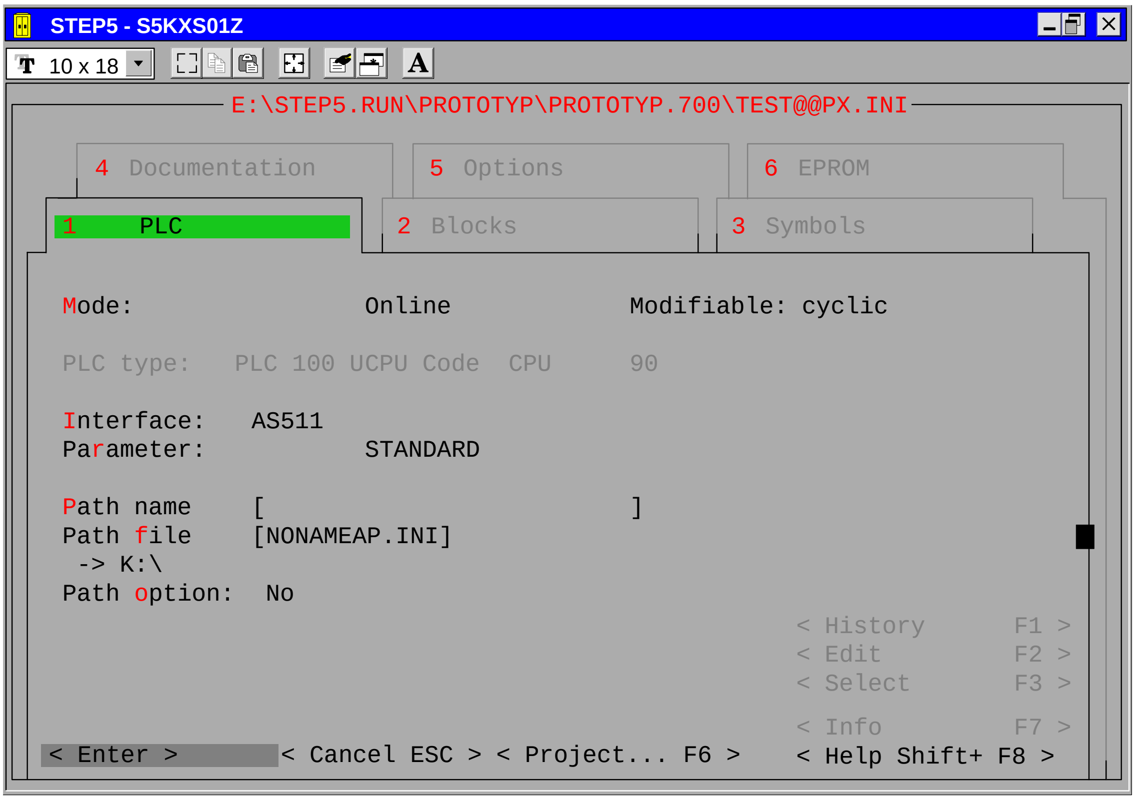


Figure 3-6 Project Settings Tab Pages

3.8 Selecting Files and Directories

Function

In this dialog box, you can select files (file list box) or directories (directory list box) by navigating through drives and into directories. The structure of both list boxes is identical.

If the job box expects a directory name (directory list box), you can select DOS directories. After exiting the dialog box, the selected directory name is entered but not file names you may have selected.

This dialog box provides additional support for certain file types that is explained in the Help for the corresponding list box.

How to Select Files and Directories

You can move between the fields with the ***TAB*** (forwards) or ***Shift+TAB*** (backwards) keys. You can obtain general information about working in file list boxes by pressing the ***SHIFT + F8*** key or by clicking the **<Help *Shift+F8*>.** The help text also provides information about the individual elements and how to use them.

You can obtain additional information about a selected input box by pressing the ***F7*** key or by clicking ***<Info + F7>.***

File

In this text box, you can enter a file name or a search mask for file names. If the end of the file name is fixed, for example ST.S5D), this ending cannot be modified.

In this box, you can also specify a drive letter or a directory path. After pressing the return key, the information is entered and the display in the file list and Dr/ directory list is updated. Using question marks as place holders, you can also enter a so-called *search mask*.

Search Mask

If you enter a *search mask* in the *File* box using question marks ???, you update the file list when you press the return key.

The search mask is displayed again when the list is displayed so that you can check the entry. As long as the search mask is active, in other words no single file name has been selected, you can only cancel the file list box.

If the job box from which the file selection was started permits question marks in the file name box, the file list box can also be exited with question marks in the *File* box.

File List

In this section of the dialog box, a list of all the files in the selected directory path is displayed. This display also depends on a *search mask* if a mask has been specified.

If you change to this list box, either a green or blue cursor appears. The blue cursor means that no element has yet been selected in the list and that no file name has been specified in the *File* text box. Otherwise the cursor is green. When you enter an alphanumeric character (number or letter), the cursor moves to the next file that starts with this character, if one exists.

Dr/directory

Using this list box, you can navigate through the DOS drives and directories, in other words you can change the current DOS directory path. If you type in an alphanumeric character (number or letter), the cursor moves to the next directory that starts with this character, if one exists.

Sorted The file list can be sorted upwards or downwards according to the names, time (date and time) and size. The Dr/directory list can be sorted in ascending and descending order.

Note

When the cursor is in the file or Dr/directory list, if you type in an alphanumeric character, the cursor jumps to the next element beginning with this character if such an element exists.

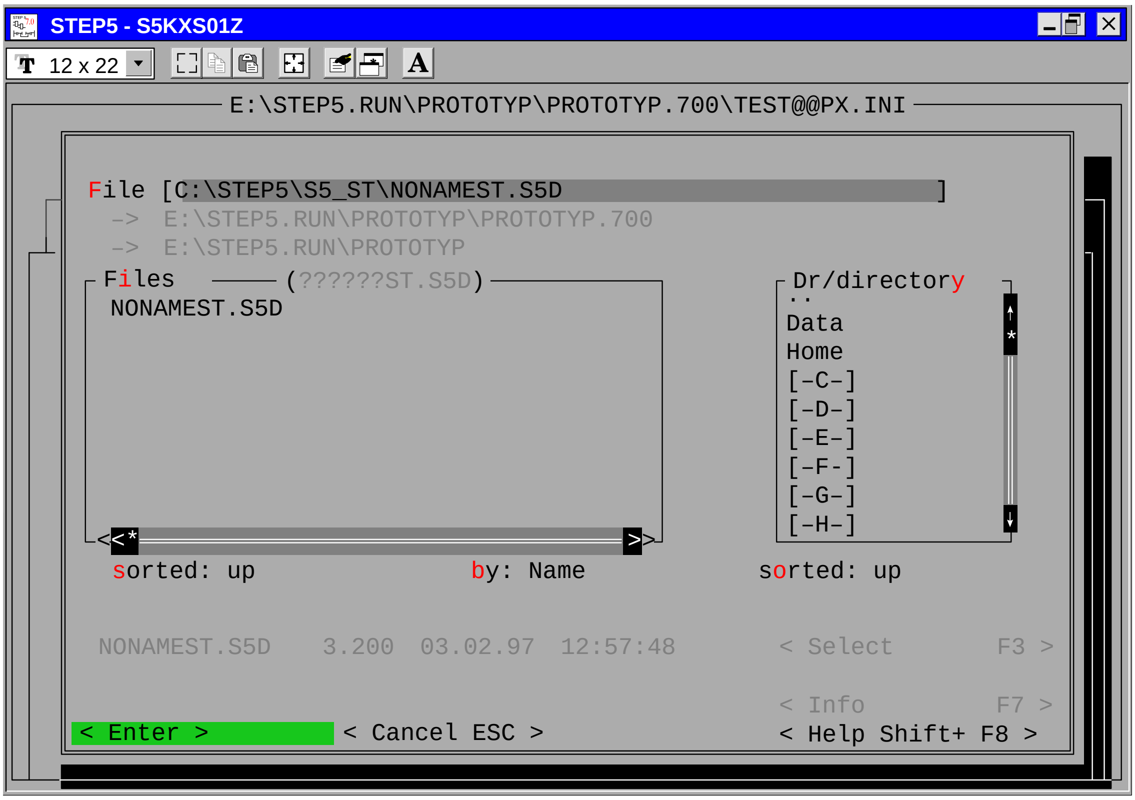


Figure 3-7 File/Directory List Dialog Box

3.9 Selecting Blocks

Function

With this dialog box, you select blocks. The block list box is displayed by pressing the **F3** key or clicking **< Select F3 >** in a text box for blocks. You can obtain information about the input options by pressing the **F7** key or by clicking **< Info F7 >** .

Using the block selection function, simplifies your input and restricts it to the blocks actually used. You can move from one field to the next in the block list box using the **TAB** or **Shift+TAB** keys. The following elements are available:

- Block
- Block list
- Block types

Working in the Dialog Box

You can change from one field to the next using the ***TAB*** (forwards) or ***Shift+TAB*** (backwards) keys. Information about working generally in the dialog box is displayed if you press the ***SHIFT + F8*** key or click the ***< Help Shift+F8 >*** button. This also provides you with information about individual elements and how to work with them.

You can obtain additional information about a selected text field by pressing the ***F7*** key or clicking ***< Info F7>***

Block

You can enter a block name in this input box. You can use all block types that are visible in the *block type* list. This list of permitted block types depends on the dialog box in which the display of the block list was activated.

Block List

This is a list of all the existing blocks (in the program file or on the PLC) whose type matches the currently set block type. When you change to this list box, either a green or blue cursor appears. The blue cursor means that no element in the list has been selected and that no block name has been specified in the upper “Block” input box. Otherwise the cursor is green.

Block Types

This list of block types displays the currently permitted types. After you select a block type with the mouse (double-click) or with the return key, the content of the block list is updated.

Note

When the cursor is in the file list or in the Dr/directory, if you type in an alphanumeric character, the cursor jumps to the next element that begins with this character if such an element exists.

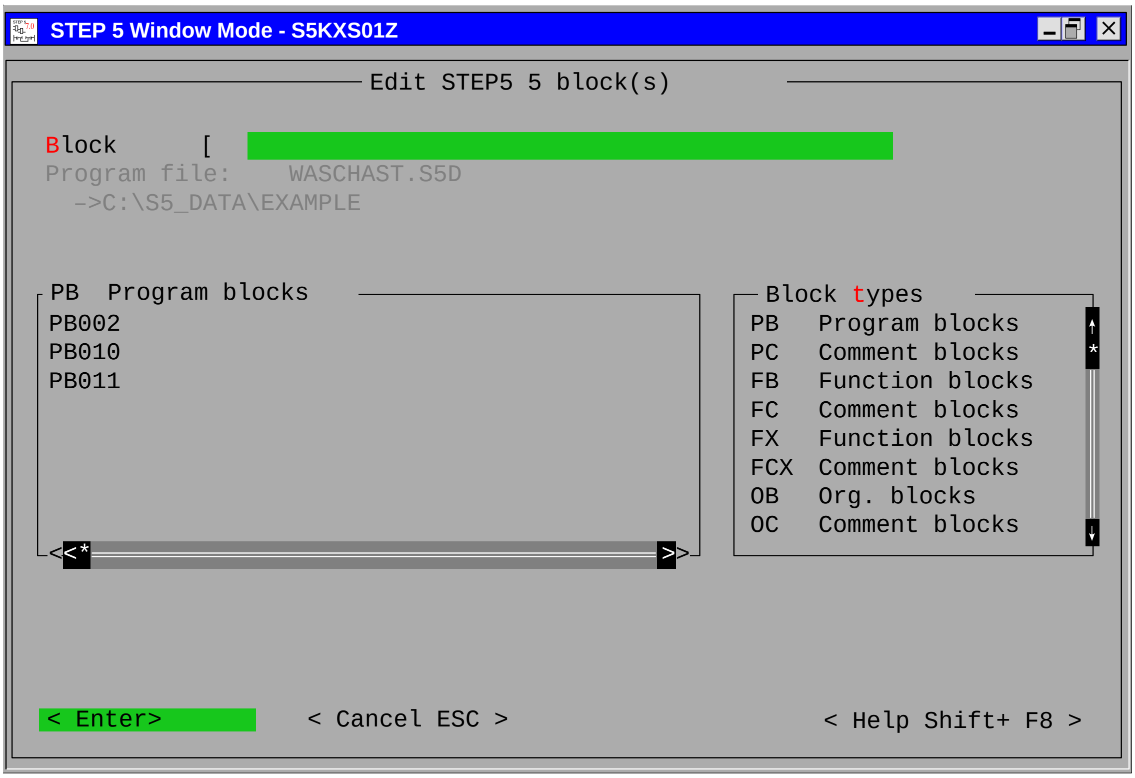


Figure 3-8 Block List Dialog Box

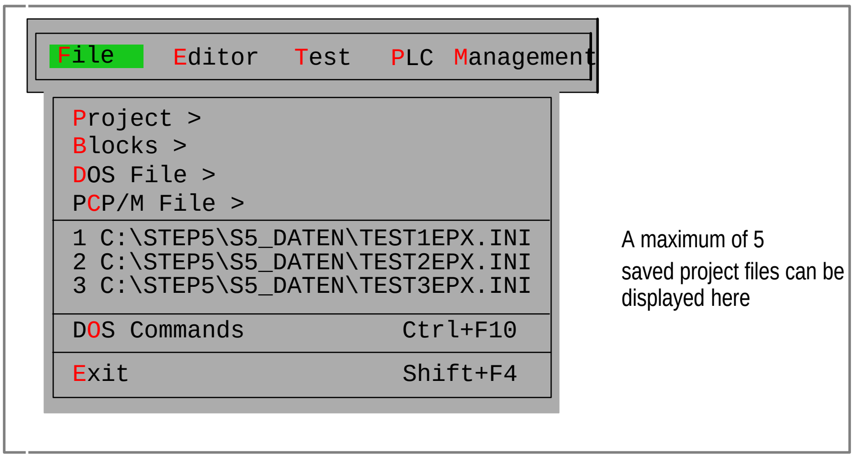
4 Creating and Handling Projects

Overview

Projects represent the entire data and programs for an automation task. They are used to save the data and programs that result from programming an automation task. The main object in creating a project is to provide the data and to write the programs.

Note for the Reader

The first part of this chapter describes the essential components that make up a project. You should first familiarize yourself with the most important objects in a STEP 5 project based on this chapter. The second part of the chapter explains the main aspects of handling the objects of a project, for example copying, transferring, comparing and deleting.



Chapter Overview

4.1 Project Settings

Overview

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 (*PX.INI) of which you can make copies. With this, you have a list 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.

Project structure

Figure 4-1 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 S5_SYS directory and then in the S5_HOME directory after they have been modified.

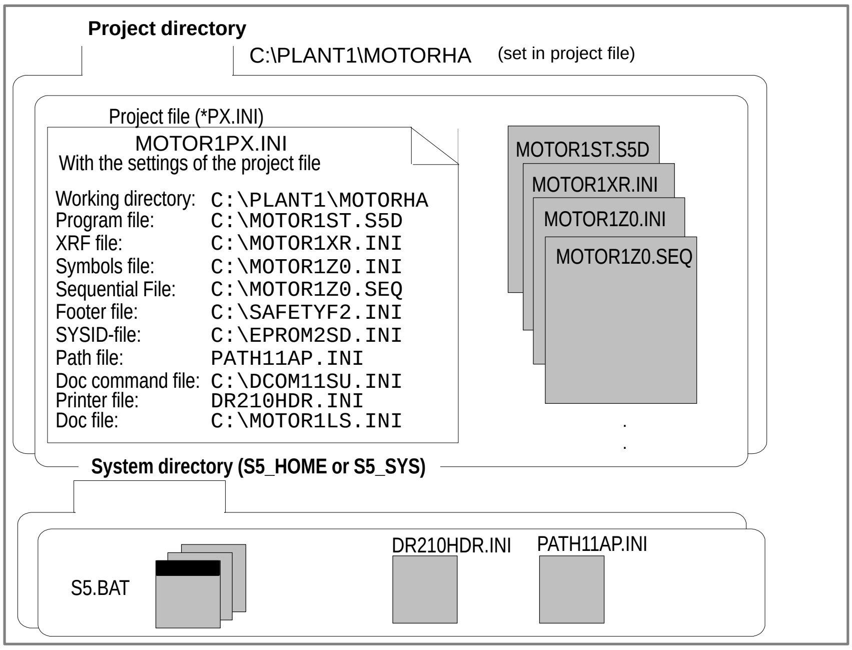


Figure 4-1 Organization of the Project File and Program File

The project structure shown here is only one of several different possibilities (refer to the Options tab in Section 4.1.1).

Functions

File

`Project >`

The following functions are available in the **File** menu.

- **File ▶ Project ▶ Set F4.** You can set all the parameters required for a specific project. These include the following:
  - Files belonging to a project. These files are always set in the job and list boxes or dialogs in which they will be addressed.
  - Parameters, e.g. symbols, method of representation (LAD, CSF, STL), character set etc. Once you have selected the settings for a project, you can only edit this project.
- **File ▶ Project ▶ Load... F10**. All the settings for the selected project are loaded. Once the project is loaded, only the files belonging to this project can be selected for editing.
- **File ▶ Project ▶ Save**. All the settings are saved in the file for the specific project.
- **File ▶ Project ▶ Save as...**. All the settings are saved in a selectable (new) file for the specific project.

Note

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

By activating **<Edit F2>** in the *Blocks*, *Symbols* and *Documentation* tab pages, you can call the relevant editor directly.

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 bold print and the name *EXA409*.

4.1.1 Project Settings

File

`Project >`
`Set F4`

Before beginning with the actual programming, you select all the parameters required for a project in the displayed tab pages. Select the menu command **File ▸ Project ▸ Set F4**. The dialog box as shown in Figure 4-2 is opened.

The dialog box is organized in tabs (Figure 4-2 shows the PLC tab page). The selected parameters (for example file names) are later entered automatically in the job or list boxes.

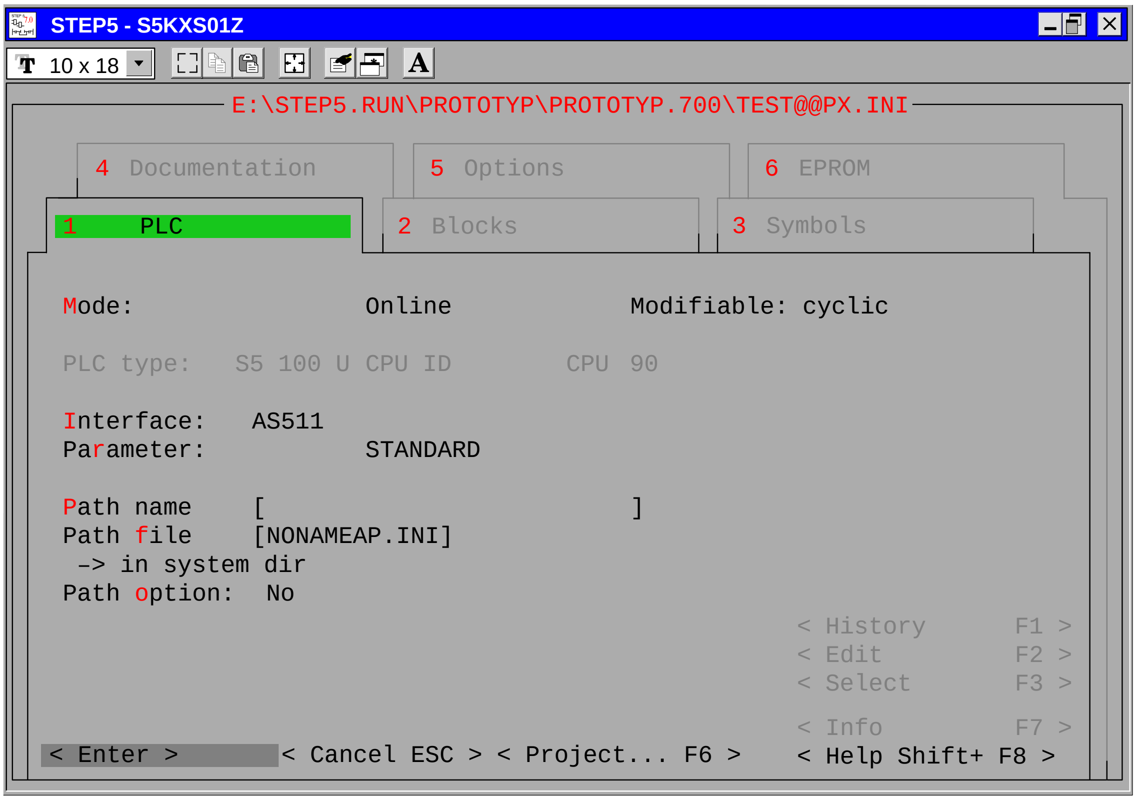


Figure 4-2 Project Settings (Tab 1, PLC)

Operation

You can move to the input files either using the ***cursor*** keys or using the ***TAB*** key (forwards) and **SHIFT+TAB** (backwards). For further information, refer to Section 3.7 or call the online help with the **SHIFT + F8** key or the **<Help Shift F8>** button.

PLC Tab

```
Mode:                    Online       Modifiable: cyclic

PLC type:   S5 100 U CPU ID       CPU   90

Interface:               AS511
Parameter:               COM1: STANDARD

Path name     [                            ]
Path file     [NONAMEAP.INI]
 -> in system dir
Path option:  No
```

`Mode:`

`Offline`	No connection to the PLC.
`Online`	Establishment of a permanent connection to the PLC. The user programs (blocks) can be tested and edited 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 established, the message *`PLC timeout`* appears. If the PG-PLC connection is interrupted, the PG is only operational again when the monitoring time set has elapsed.
`Dynamic`	This mode is only possible when there is a connection via a bus path. The connection is only established when access is required and it is terminated again as soon as access is complete.

`Modifiable`

You can select whether and how a program can be modified on the PLC. With the ***F3*** key, a box is displayed with the possible modification modes in which you can set the following:

`No`	You cannot modify a program on the PLC.
`Stop`	You can only modify a program on the PLC when the PLC is in the stop mode.
`Cyclic`	You can modify a program on the PLC during the processing cycle.

`PLC type:`

If there is a connection to the PLC, the type of PLC is displayed here.

`Interface:`

If you press ***F3,*** various interfaces are displayed from which you can select one. The AS 511 interface is the default. With these interfaces, you can select the mode directly. If you select a different interface, a bus path must be edited before you can select the mode.

`Parameter:` In conjunction with the activated interface, the following settings are possible:

Standard: Default for the particular interface

For AS 511: `COM 1 to COM 4` and additional special drivers 1 to 7.

With this parameter, you can assign values (wait times, repetition times etc) for the H1 or L2 interface. The selectable parameters are read for H1 from the S5@@@@H1.INI and for L2 from the S5@@@@L2.INI file.

For the AS511 interface, the parameters of the files AS511S01.DAT to AS511S07.DAT were included in the S5@@@@AS.INI file.

`Path name` Name under which an edited path (Section 12.1 *Bus Paths*) 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 S5_SYS\AP.INI as a template and after modification it is located in the S5_HOME directory. If you create a new AP.INI this is always stored in S5_HOME.

`Path option`

`No` Files assigned to a bus path are not entered.

`Confirm` If files are assigned to a bus path, and if the path is set, the files are only entered globally in the settings 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 settings without user confirmation.

Blocks Tab

```
Program file      [NONAMEST.S5D                         ]      RW
 -> C:\STEP5\S5_Daten

X reference list  -NONAMEXR.INI
 -> C:\STEP5\S5_Daten

Representation:   LAD
STL addresses:    WORD

[ ] with comments
[ ] with checksum
```

`Program file` You can assign any name to the file, the extension is ST.S5D. All S5 blocks are managed in this file.

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.

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.

`File mode` Selectable file mode:

`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 being written to. A different S5 system is accessing the file. Once the access is complete, this entry is cleared.

RO: Read only.

`X reference list` The name of the file (*XR.INI), which will contain the cross reference list, is only displayed here and cannot be modified. For information on creating the XRF file, refer to Section 17.1 *Management, Generate XRF*

`Representation` You can select between one of the three methods of representation, LAD, CSF, STL.

`STL-addresses` When editing in STL, the relative command addresses are displayed as follows:

`WORD` In words

`Byte` In bytes

`[x] with comments` Comments are also displayed.

`[ ]` The line comment, segment comment and segment title are also displayed.

`[x] with checksum` When you read blocks from the PLC, the checksum is used to check the transfer

If you press the **F2** key or click the **<Edit F2>** button, you can call the editors directly. The confirmation and updated cross-reference list and assignment list options are taken from the job box.

Symbols Tab

```
Symbols file      [NONAMEZ0.INI                       ]      RW
 -> C:\STEP5\S5_Daten

Assignment list   [NONAMEZ0.SEQ                       ]      RW
 -> C:\STEP5\S5_Daten

Symbol length     [8 ]
Comment length    [24]

[ ] Display symbolic
[ ] Operands symbolic
```

`Symbols file`[1]	The name of the symbols file (*Z0.INI). If you set this file, then providing you have set *`Display symbolic`*, you work with symbolic operands (in the editors and in documentation output). This means that symbols and symbol comments are assigned to the absolute operands. You create this assignment with the symbols editor. As soon as you set this file, the setting for the sequential file is made automatically.
`Assignment list`[1]	The source file (*Z0.SEQ) which contains the assignment list is set as soon as you have named the symbols file. This is the file that you edit with the symbols editor. On completion of editing, the symbols file is generated. If there is no assignment list, it can be recreated from the symbols file.
`Symbol length`	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 file.
`Comment length`	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 the symbols file assigned (*Z*.INI).
[x] Display symbolic [ ]	The input and output of symbolic operands is possible. To do this, you must however specify a symbols file. The input and output of symbolic operands is in absolute format.
[x] Operands symbolic [ ]	Operands are displayed symbolically. If symbols are longer than 8 characters, they are truncated (only in LAD, CSF). Operands are displayed in absolute form. Symbols are displayed in line 3 in LAD/CSF. In STL they are displayed in absolute and symbolic form.

1) The two files ZO.INI, ZO.SEQ can be selected. As soon as one of these two files is set or selected, the other is updated (in other words the two files have the same name except for the ending).

If you press the **F2** key or click the **<Edit F2>** button, you can call the editors directly. The confirmation and updated cross-reference list and assignment list options are taken from the job box.

Documentation Tab

```
Footer file  [NONAMEF1.INI                              ]
 -> C:\STEP5\S5_Daten
Doc command file  [NONAMESU.INI                         ]
 -> C:\STEP5\S5_Daten
Printer file [NONAMEDR.INI]
 -> im Systemkatalog

Printer interface: from printer file
Character set: ASCII                     Footer: No

   Documentation to
   (X) Printer
   (X) File     [NONAMELS.INI]
        -> C:\STEP5\S5_Daten
```

`Footer file`	The name of the footer file. A footer is stored in this file. The footer is created with the footer editor and automatically printed out at the end of a page if you select a size for the footer. This footer is output automatically when you document your work. Depending on the entry you make in the **Footer** input field, **F1.INI** for an 80 character wide footer or **F2.INI** for a 132 character wide footer is entered.
`Doc command file`	In this file, you can store commands for creating extensive documentation with KOMDOK. Refer to editing doc commands or editing the structure *(Section 18.4).*
`Printer file`	This file must contain the control characters of your printer for setting the pitch: It also includes the following parameters: • Format (A4/A3) • Lines per page • Optional parameters The file is available as a template in the S5_SYS\AP_INI directory. If you edit the template, the file is copied to the S5_HOME catalog and the changes are made there. If you create a new DIR.INI file, this is always stored in the S5_HOME catalog. If no file is specified, the parameters of the PT88 apply.
`Printer interface`	from printer file: The printer port is read from the printer parameters (DR.INI). LPT 1, LPT 2, LPT 3 You can select the printer port, these settings do not affect the printer parameters (DR.INI).
`Character set`	Only valid for enhanced output (see Section 18.1). The following can be selected: ASCII: Documentation is only printed with the characters of the ASCII character set e.g.: !—][——————( )—]! SEMI GRAPH.: Documentation is printed with the IBM character set e.g.: |—][——————( )—|!

`Footer`

`No:`	No footer is printed out with the documentation.
`80:`	A 80 character long footer is printed
`132:`	A 132 character long footer is printed

Documentation to

`[x] Printer`	Documentation on printer
`[x] File` `[**.INI]`	All printouts are written to the file (*LS.INI) specified here. If this file already exists, the new data are appended.

1) The two files ZO.INI, ZO.SEQ can be selected. As soon as one of these two files is set or selected, the other is updated (in other words the two files have the same name except for the ending).

If you press the **F2** key or click the **<Edit F2>** button, you can call the editors directly. The confirmation and updated cross-reference list and assignment list options are taken from the job box.

Options Tab

```
Project directory [NONAMEST.S5D                    ]
 -> C:\STEP5\S5_Daten

on exiting STEP 5/ST:
     [X] Confirm always
     [X] Save project settings
     [X] Note active optional package


[ ] Warnings if incompatible with V 6.x
```

`Project directory`	By specifying this DOS path you set the paths for all files in the project settings to the same path (except for *AP.INI and *DR.INI). If the individual paths of the files are different, no path is displayed in this box.
`[X]Confirm always`	You are prompted for confirmation whenever you exit STEP 5.
`[X]Save project settings`	If you activate this setting, modified project settings are automatically saved in the selected project file when you exit STEP 5. If no project settings were modified, the settings are not saved.
`[ ] Note active optional package`	If you activate this setting, any optional package (GRAPH 5, COM 155H, COM 95F) active when you exit STEP 5 is started automatically when you restart STEP 5.
`[ ]Warnings if incompatible with V 6.x`	If you select this setting, the program automatically checks whether DOS paths you have selected are within the restrictions imposed by STEP 5 Version 6.x. These are as follows:

– Drives A: to J: for the program file
– Drives A: to P: for other files
– For each drive a maximum of one directory can be used.

If these criteria are not met, STEP 5 displays a warning. With this setting you can make sure that the files you have currently set are compatible with project settings for Version 6.x.

EPROM Tab

```
Prommer type: internal

SYSID file    [NONAMESD.INI                                    ]
  –> C:\STEP5\S5_Daten

Storage mode: WORD
```

`Prommer type` — Select by double-clicking or with the ***F3*** key.

- `no` — No prommer being used
- `internal` — The internal prommer is used
- `external LPTn` — An external prommer is used via the parallel port.

`SYSID file` — Contains the system identification, selected in a file list box, by double-clicking or with the ***F3*** key.

With the SYSID OUT function, the SYSID blocks found in the submodule are automatically stored in the SYSID file. With the SYSIDINP function, the block in the SYSID file is written to the submodule starting at address 0.

`Storage mode` — This setting decides how the data (programs and data blocks) are stored on the EPROM. You can select the type of storage by double-clicking or with the ***F3*** key.

- `WORD` — Write/read word-oriented, for example S5-135 and S5-150 (all types)
- `WORD/BLOCK` — Write/read byte-oriented, for example for S5-155U (all types)
- `BYTE` — Mandatory for the CPU 946/947 (memory module 355). For the S5-155H, the first character of the user data of a block is at the paragraph boundary (16th byte).

4.1.2 Load Project

File
Project >   Load ...F10

With this function, you load the settings you selected under **File** ▶ **Project** ▶ **Set** and saved in a *PX.INI file (see Section 4.1.1). 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 PX.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.

Select the menu command **File** ▶ **Project** ▶ **Load...F10** The Load project settings job box is displayed. Here, you can select a *PX.INI file. After selecting **Load**, all the settings are loaded from the *PX.INI file.

4.1.3 Save Project

File
Project >   Save

With this function, you save the current settings you have made under **File** ▶ **Project** ▶ **Set** (see Section 4.1.1). The settings are saved in the currently selected *PX.INI file.

With **File** ▶ **Project** ▶ **Save** a message box is displayed in which you decide whether or not to save the settings.

4.1.4 Save Project As

File
Project >   Save As ...

With this function, you save the current settings you made under **File** ▶ **Project** ▶ **Settings** (see Section 4.1.1). The settings are saved in a *PX.INI file that you select.

With **File** ▶ **Project** ▶ **Save As** the *Save project settings* job box is displayed. Here, you can select a *PX.INI file or create a new one.

4.2 Managing Blocks

File
`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
- check and compress blocks in the program file

4.2.1 Block Directory

Overview

The following directories can be output:

From the current program file:

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

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

Note

When you display blocks on the screen, you obtain a further job box in which you can branch directly to the editor by selecting one of the displayed blocks (see screen display).

File

```
Blocks >
  Directory...F3
```

If you select the menu command **File ▸ Blocks ▸ Directory...F3,** the *Blocks - Directory: Settings* job box is displayed. Here you can navigate through the job box and select blocks (*see Section 3.9).* Once you have set a block and clicked the **< Output >** button, the directories of selected blocks are output on the required output device.

If you select screen display, you can also branch to the editor with the **F2** key.

Input	**Explanation**
Directory of	
`(x)Program file`	This field displays the currently selected program file. You can edit this name or replace it with an existing file name from the file list box using the ***F3*** key.
`(x)PLC`	Displays the PLC on which the block is stored. The PLC is selected in the project settings (see Section *4.1.1*) and this is only possible in the online mode.
Selection	
`Block list [        ]`	Here you select the blocks. You can specify blocks in absolute or symbolic form (or as a mixture of both). If you want to edit an existing block or want to display the currently permitted block types, press the ***F3*** key or click the ***< Select F3 >*** button. STEP 5 displays a list of the currently possible inputs if you press the ***F7*** key or click the ***< Info F7>*** button.
Output to	
`(x) Screen`	The directories are displayed on the screen.
`(x) Printer`	The directories are logged on the printer.
`(x) File [     ]`	The directories are written to a selectable file. A file list box is displayed when you press the ***F3*** key or double-click the input box.
Options	
`Printout type:`	You can select the type of printout by double-clicking the input box or by pressing the ***F3*** key.
`[x]mixed with preheaders`	If you select this option, the preheaders of the blocks are also output.
`[x]FBs with name`	If you select this option, FBs and their names are printed out.
< Output >	The PG transfers the selected blocks. If errors occur, various alternatives are displayed in list boxes in which you can make your selections.

Examples of Input

`Block list [                ]` you specify blocks either in absolute or symbolic form, or a mixture of both.

Single Block

Single block, in absolute or symbolic form.

```
[PB100                                  ]
[DX 14                                  ]
[OB 10                                  ]
[FCX 231                                ]
[–Plant1                                ]
```

Block List

List with a maximum of six single blocks. The blocks are separated by commas. If the comma follows a symbolic name, the comma must be preceded immediately by "\" to delineate the symbol.

```
[PB100 , PB123                          ]
[–Plant1\, –Plant2                      ]
[–Plant1\, FB45, –Plant2\,–Control      ]
```

Block Range

A range indicated by two single blocks. The blocks are separated by a hyphen. If the hyphen follows a symbolic name, the hyphen must be preceded immediately by a "\" to delineate the symbol. Both blocks of the block range must be the same type, the first block number must be lower than the second block number.

```
[PB100 – PB123                          ]
[ –Plant1\- –Plant2                     ]
[–Plant1\- FB45                         ]
```

Block Type

You can specify one or all block types

```
[PB          ]all program blocks
[A           ]all blocks (but not the DOK block)
[OC          ]all OB comments
[DB          ]all data blocks
[#           ]all DOC blocks
```

DOC Block

Block preceded by the # character

```
[#MOT_P                                 ]
[#DBDO.003                              ]
[OC#OBDO.024                            ]
```

Output to Screen

The blocks are displayed on the screen in a separate job box:

Block-directory-program file: output

To edit or modify the list, follow the steps below:

1. Select a block from the list.
2. Press the **F2** key or click the **< Edit F2 >** button. STEP 5 then opens this block in the appropriate editor window (fast jump to the editor).

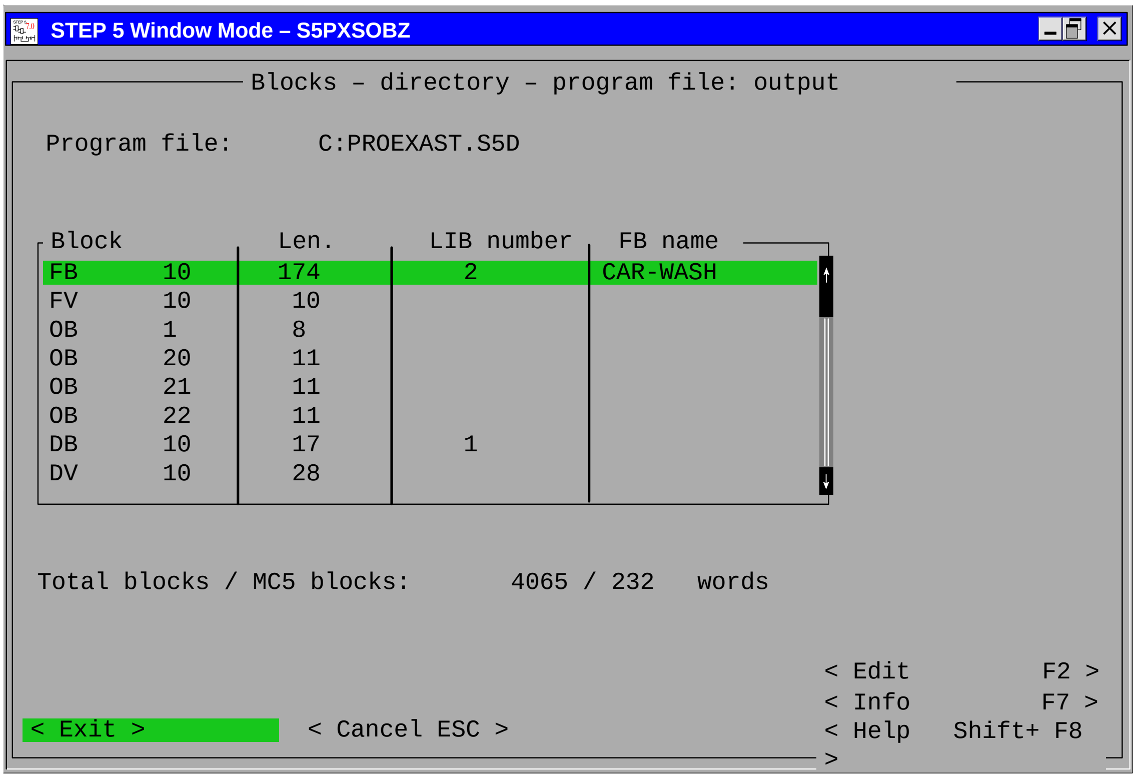


Figure 4-3 Blocks-Directory-Program file: Output

1) The two files ZO.INI, ZO.SEQ can be selected. As soon as one of these two files is set or selected, the other is updated (in other words the two files have the same name except for the ending).

If you press the **F2** key or click the **<Edit F2>** button, you can call the editors directly. The confirmation and updated cross-reference list and assignment list options are taken from the job box.

4.2.2 Copy (Transfer) Blocks

Overview

With the *Transfer* function, you can copy blocks from the programming device to the PLC and vice-versa, as follows:

- a range 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 blocks
- the entire program file
- from the selected program file to a selectable drive and selectable program file (**file - file**). Both files can be selected.
- 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**).

Note

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 on FD?
```

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.

File
`Blocks >`   `Transfer ...F5`

Select the menu command **File ▸ Blocks ▸ Transfer..F5.** The *Transfer blocks(s)* dialog box is displayed. Here, you can navigate to directories and select single blocks (*User interface see Section 3.9)*

Note
The transfer of blocks depends on the particular PLC being used. This means that not all blocks that are displayed can actually be transferred.

Only blocks up to a maximum of 4 Kwords (8 Kbytes) can be transferred.

The blocks are transferred to the PLC in the following sequence: SB, PB, FB, FX, OB, DB and DX.

Inputs	Explanation
Transfer from `(x)Program file`	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 list box using the ***F3*** key.
`(x)PLC`	Specifies the PLC on which the block is stored. This entry is made in the project settings and is only possible in the online mode (*see Section 4.1.1*).
to `(x)Program file`	The program file name is displayed in this field. You can edit this name or replace it with an existing file name from the file list box using the ***F3*** key.

Selection
`(x)Block list`
`[    ]`
	After selecting this parameter, you can enter your block selection in absolute or symbolic form (or a mixture of both) in the following input field. You can display currently permitted block types with the ***F3*** key. You can display information about the entries in the block list field with the ***F7*** key. Input examples are given in Section 4.2.1.

`Block` `[    ] to [    ]`	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 (for example **PB7**) and the new block name (for example **PB22**) in the *to* [ ] field. When you copy blocks you must not change the block type. You can obtain more information with the ***F7*** key.
`(x)Entire file`	With this option, you can select the whole program file (including docfiles).
Option `[x]Confirm before overwriting`	Modifications are only entered when you save after you confirm a system prompt.
<Transfer>	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.

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.

4.2.3 Compare Blocks

Function

With this function, you can compare 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 blocks 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.

```
File
  Blocks >
    Compare ...F6
```

Select the menu command **File ▸ Blocks ▸ Compare..F6** or press the **F6** key. The *Compare block(s)* dialog box is displayed. Here, you can browse through the box and make your selection (*see Section 3.9).*

Inputs	Explanation
Compare	
`(x)Program file`	The preset program file is displayed in this field.
`(x)PLC`	Specifies the PLC on which the block is or will be stored. This entry is made in the project settings (*Section 4.1.1*) and is only possible in the online mode.
with	
`(x)Program file`	The program file name is displayed in this field. You can edit this name or replace it with an existing file name from the file list box using the ***F3*** key.
`(x)PLC`	Specifies the PLC on which the block will be stored. This entry is made in the project settings (*Section 4.1.1*) and is only possible in the online mode.

Selection
`(x)Block list`
`[    ]`
	After selecting this parameter, you can enter your block selection in absolute or symbolic form (or a mixture of both) in the following input field. You can display currently permitted block types with the ***F3*** key. You can display information about the entries in the block list field with the ***F7*** key.

`Block` `[    ] with [    ]`	Here you specify single blocks that you want to compare.
Output to	
`(x) Screen`	Output is displayed on the screen.
`(x) Printer`	Output is printed on the selected printer.
`(x) File`	Output is made to the selected file.
<Compare>	The PG compares the selected blocks.

When you compare blocks on the PLC, remember that only block types are permitted that can be selected in the job box.

4.2.4 Delete Blocks

Function

With this function you can delete the following:

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

File

```
Blocks >
  Delete ...
```

Select the menu command **File ▸ Blocks ▸ Delete......**The job box *Delete block(s)* is displayed. Here, you can browse through the box and make your selection (*see Section 3.9*).

Inputs	Explanation
Delete `(x)Program file`	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 list box using ***F3***.
`(x)PLC`	Specifies the PLC on which the block will be deleted. This entry is made in the project settings (*Section 4.1.1*) and is only possible in the online mode.

Selection
`(x)Block list`
`[    ]`
	After selecting this parameter, you can enter your block selection in absolute or symbolic form (or a mixture of both) in the following input field. You can display currently permitted block types with the ***F3*** key. You can display information about the entries in the block list field with the ***F7*** key.

`(x)Entire file`	By marking the field with an *X* you can select and delete the whole program file (including docfiles).
`(x)Delete entire PLC`	All blocks on the PLC are deleted (only in the STOP mode). The PLC sets defined start statuses in its memory (RAM). For more information, refer to the programming instructions for the particular PLC..
Option `[x]` Confirm before deleting	The delete function is only started after you confirm a prompt.
<Delete>	The function is executed.

4.2.5 Compress Blocks

File

```
Blocks >
  Compress
```

Select the menu command **File ▸ Blocks ▸ Compress.** This function eliminates gaps in the program file that result from deleting or reloading blocks. The STEP 5 blocks in the program file are checked and compressed. If any error occurs, this is displayed.

During the checking process you can establish whether the structure of the program file is really correct, or whether it has been damaged by a power cut or a system crash during saving.

Files which are 0 Bytes long are also registered as faulty.

4.3 DOS File

File
DOS File >

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

- Display 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.

Using the Functions

You select the files from a file list box assigned to each menu command. The structure and use of the list box is the same as for all functions and is described in Section 3.8.

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.

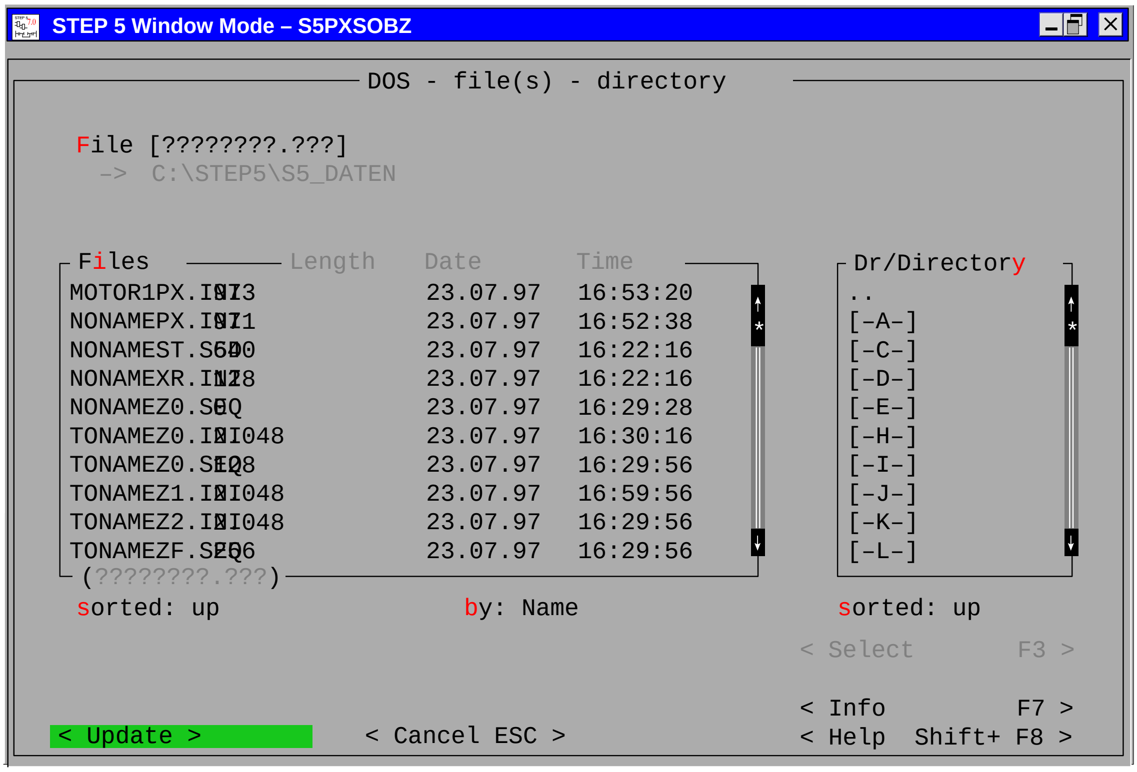


Figure 4-4 File List box for DOS Files

4.3.1 Display a Directory

Function

This function displays a list with the directory or directories of one or more files.

File
DOS File >
Directory ...

Select the menu command **File ▸ DOS File ▸ Directory** or press ***Ctrl + F7***. The *DOS files directory* job box is displayed. Here, you can browse through the box and make your selection. Depending on the entries you make, a list of files is displayed.

Inputs	Explanation
`File`	The file name marked by the cursor in the list of field names is displayed here. 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 after you click the **<Update>** button or press the **Insert** key.
`Dr/ directory`	Here, you can select a drive and a directory on this drive. Once you have made the selection, the content of the directory appears in the display field.

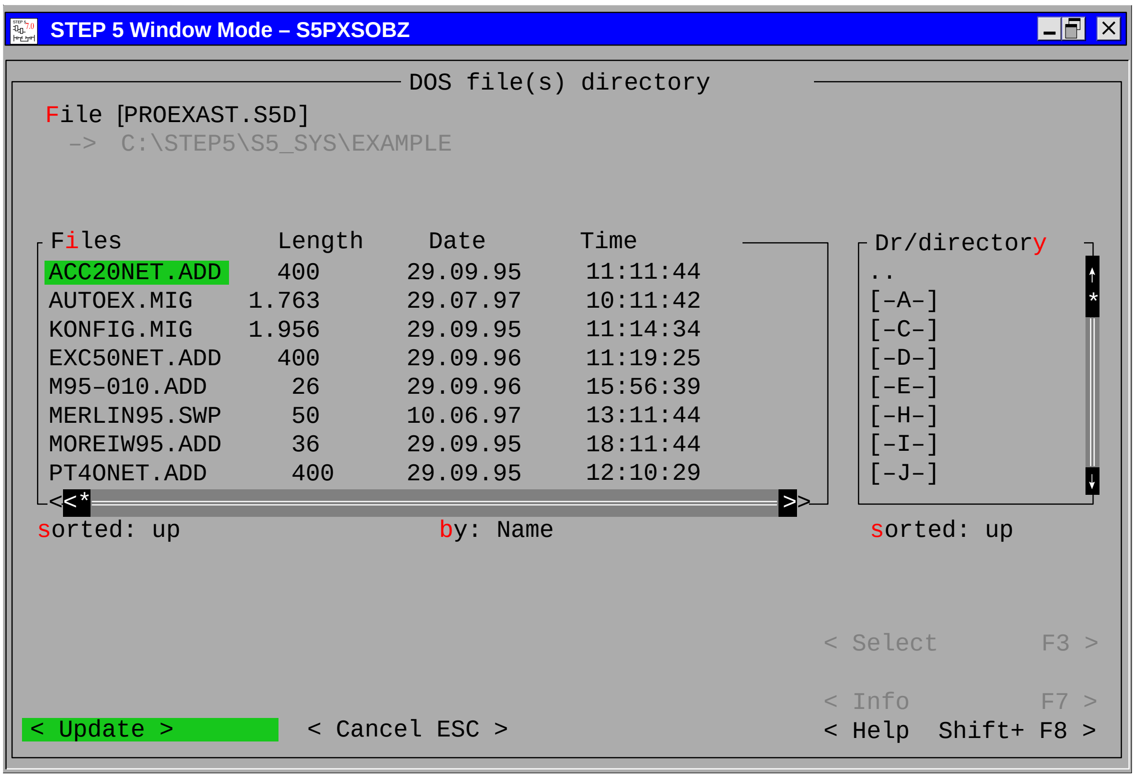


Figure 4-5 DOS Files Directory

4.3.2 Copy DOS Files

Function

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 (only with single files).

File

```
DOS File >
  Copy ...
```

Select the menu command **File ▸ DOS File ▸ Copy** or press ***Ctrl + F8***. The function copies one or more files between different drives or directories.

`Source file  [ ]`	Name of the file you want to transfer.
`Dest file    [ ]`	Name of the file transferred
	For information about using the function press the ***F7*** key or click ***< Info F7>***.
`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 only displayed if question marks (or *.*) are entered in the *Source file* field.
`<Copy>`	The function is executed.

Procedure

Follow the steps below:

1. Select the drive and directory in the *Source dr/dir* field from which you want to transfer (copy) one or more files.
2. You can either transfer single files or all the files listed in the *Source files* field.

   Single files: 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.

   Several files: 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.
3. 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.
4. Click *<Copy>* to start the copy function.

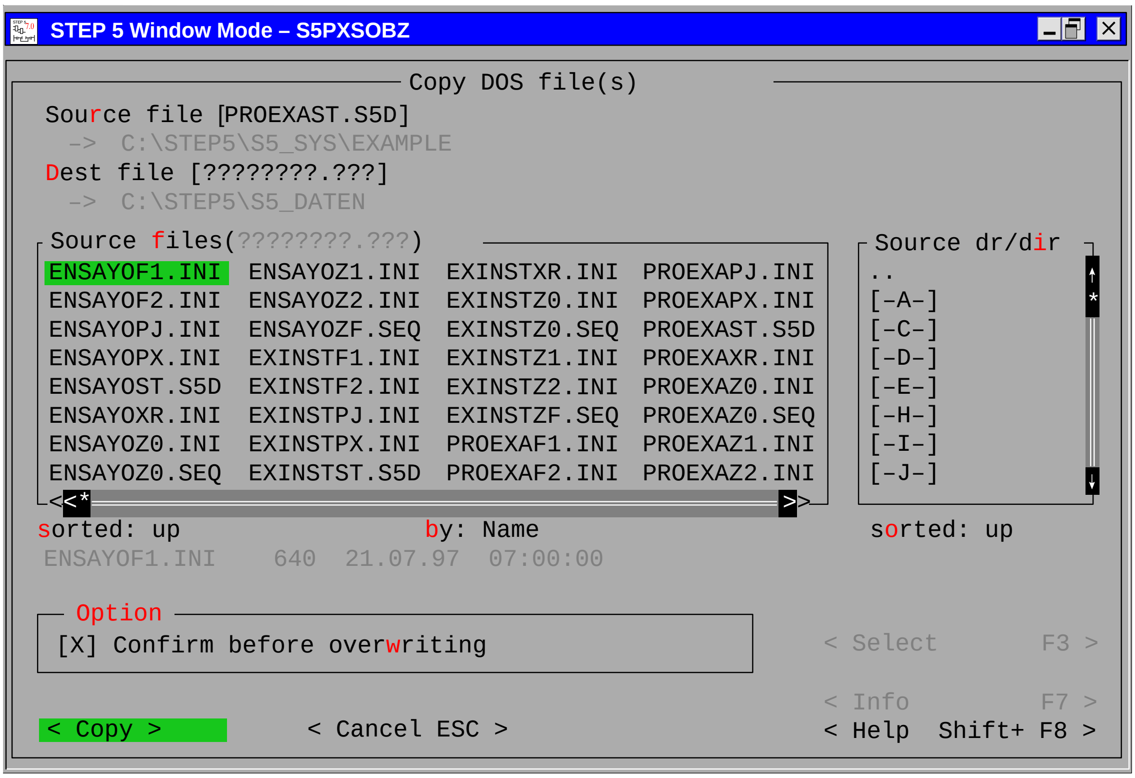


Figure 4-6 Copy DOS File(s)

4.3.3 Delete DOS File

Function

This function deletes files (one or all) in a selected directory.

File

```
DOS File >
  Delete ...
```

Select the menu command **File ▸ DOS File ▸ Delete**

The *Delete DOS file(s)* job box is displayed. You can browse through this box and make your selection (see Section 3.6).

4.4 PCP/M File

File
PCP/M File >

The following functions are available:

- Output directory of PCP/M files from selectable USER areas
- Conversion of PCP/M files to S5-DOS ST/MT files. They can then be run and edited under the S5-DOS operating system .
- 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 edit them under the PCP/M operating system.

  Here, you have functions available to process PCP/M media. PCP/M media are disks formatted under PCP/M.
- Delete PCP/M files

Using the Function

You select the file(s) in a file list box assigned to each menu command. The structure and how to work with this list box is essentially the same for all functions and is described in Section 3.8.

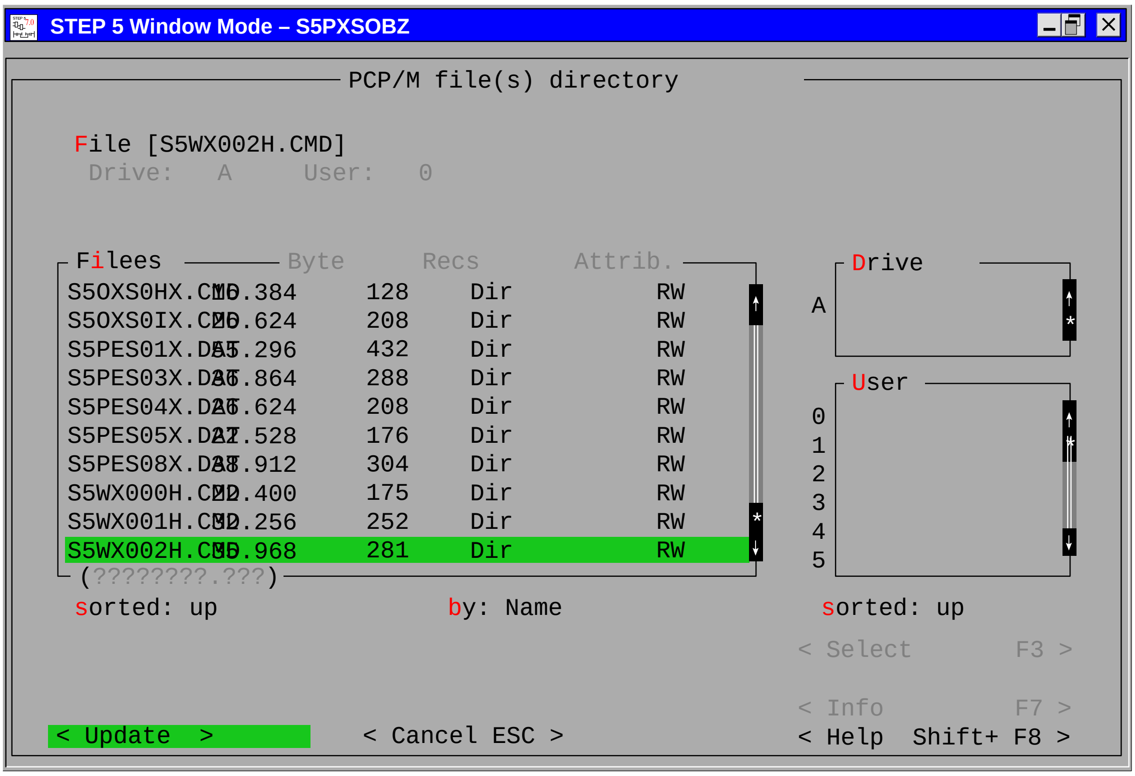


Figure 4-7 File List Box for PCP/M Files

4.4.1 Display Directory

Overview

You can display a file list of a selectable USER area from a PCP/M disk.

File

```
PCP/M File >
  Directory ...
```

Select the menu command **File ▸ PCP/M File ▸ Directory...** The *PCP/M file(s) directory* job box is displayed. Here you can browse and make your selections (see Section 3.8). Depending on your input, a directory known from PCP/M is displayed in a window:

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

File List Box

You can make the following entries:

`File`	The file name marked by the cursor in the directory is displayed here. 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`	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.
`Drive`	All the existing PCP/M drives are displayed. You can select one from this list.
`User`	List of all USER areas. You can select one of the user areas in this list.
<Update>	The function is executed

4.4.2 Copy PCP/M Files to DOS File

Overview

With this function, you can convert PCP/M files to S5-DOS files:

File

```
PCP/M Files >
  Copy
    PCP/M -> DOS
```

Select the menu command

File ▸ PCP/M File ▸ Copy PCP/M → DOS...

The *Copy PCP/M file(s) to DOS file(s)* job box is displayed. You can browse through this box and make your selection (see Section 3.8*).* Depending on our entries, a list of PCP/M files is displayed.

File List Box

Explanation of the file list box:

`Source file`	Name of the file to be transferred. If you want to find a particular file or group of files, you can enter the name here or enter a search mask with wildcards (? or *). Search mask: for example ???A*.* ??AB??.I?? If you use a search mask, the files in the *Source files* field are updated. Then only the files whose names match the mask are displayed.
`Source dr`	Here you select the drive from which the file will be transferred. This is then displayed in the *Drive* field. You can select a drive by double–clicking or with the ***F3*** key.
`Source user`	Here you select the user area of the source. This is displayed in the *User* field. You can select a user area by double–clicking or with the ***F3*** key.
`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 *Source file* field.
`Drive: –––`	Drive from which the file will be transferred. This field is only for information, no input possible.
`User: –––`	USER area containing the source. This field is only for information, no input possible.
`Dest file`	Name of the destination file. You specify **one** destination file by entering a file name without wild cards, for example `ABCDEFGH.123`. You can only specify the destination file in this way if the source file was specified as a single file without wild cards. You can specify **more than one** destination file by entering only wild cards (question mark or asterisk) in the file name for example `????????.???` or *.*. You can enter the destination file in this form when a single file or more than one file (with wild cards) was used as the source file.
`[X] Confirm before overwriting`	If you select this option, files are only overwritten after you have confirmed a system prompt.
<Copy>	The function is executed.

4.4.3 Copy DOS File to PCP/M File

Overview

With this function, you can convert S5-DOS files to PCP/M files:

File

```
PCP/M File >
  Copy
    DOS -> PCP/M
```

Select the menu command

File ▸ PCP/M file ▸ Copy DOS → PCP/ M...

The *Copy DOS files to PCP/M* job box is displayed. You can browse through this box and make your selection (see Section 3.8). Depending on our entries, a list of PCP/M files is displayed.

File List Box

Explanation of the file list box:

`Source file`	Name of the file to be transferred. If you want to find a particular file or group of files, you can enter the name here or enter a search mask with wildcards (? or *). Search mask: for example ???A*.* ??AB??.I?? If you use a search mask, the files in the *Source files* field are updated. Then only the files whose names match the mask are displayed.
`Source dr/dir`	Here you select the drive from which the file will be transferred. This is then displayed in the *Drive* field. You can select a drive and directory by double-clicking.
`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 *Source file* field.
`Dest file`	Name of the destination file. You specify **one** destination file by entering a file name without wild cards, for example `ABCDEFGH.123`. You can only specify the destination file in this way if the source file was specified as a single file without wild cards. You can specify **more than one** destination file by entering only wild cards (question mark or asterisk) in the file name for example `????????.???` or *.*. You can enter the destination file in this form when a single file or more than one file (with wild cards) was used as the source file.
`Drive: ---`	Drive to which the file will be transferred.
`User: ---`	USER area containing the destination.
`[X] Confirm before overwriting`	If you select this option, files are only overwritten after you have confirmed a system prompt.
`<Copy>`	The function is executed.

4.4.4 Delete PCP/M file

Overview

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

File

```
PCP/M File >
  Delete ...
```

Select the menu command

File ▶ PCP/M file ▶ Delete...

The *Delete PCP/M file(s)* job box is displayed. You can browse through this box and make your selection (see Section 3.8*).* The significance of the input fields is the same as described in Section 4.4.2.

Note

All files in a USER area are only displayed when question marks are entered in the *File* field.

4.5 DOS Commands *CTRL + F10*

MS-DOS Prompt

Select the menu command **File ▶ DOS Commands** or press ***CTRL+F10***. The MS-DOS system prompt is then displayed. You can now enter MS-DOS commands.

S5SHELL.BAT

The current command processor (usually COMMAND.COM) is loaded.

In STEP 5/ST you can open a DOS environment without exiting the package. You return to the STEP 5/ST package by typing in the command ”EXIT”.

If a file with the name S5SHELL.BAT is created in the home directory, this is executed when you call the DOS commands function. This, for example, allows File Managers such as the DOSSHELL (of MS–DOS 5.0) to be started.

The DOS commands should only be used when you want to perform functions with operating system tools.

Caution: Make sure that no resident programs such as DOSKEY, KEYB etc. are loaded. Also make sure that no functions that make drive assignments such as SUBST or ASSIGN are active. This also applies to logging on in a network.

Exiting DOS

Type in the command ***EXIT*** to return to the STEP 5 user interface.

4.6 Exit *SHIFT+F4*

Function

With the menu command **File ▸ Exit** or ***SHIFT + F4*** you terminate STEP 5. If you selected *Confirm always* in the project settings, you are prompted to confirm that you want to exit the program so that you cannot terminate it accidentally.

- Answer with ***Exit*** if you really want to exit STEP 5.
- Answer with ***Cancel*** if you want to return to the user interface.

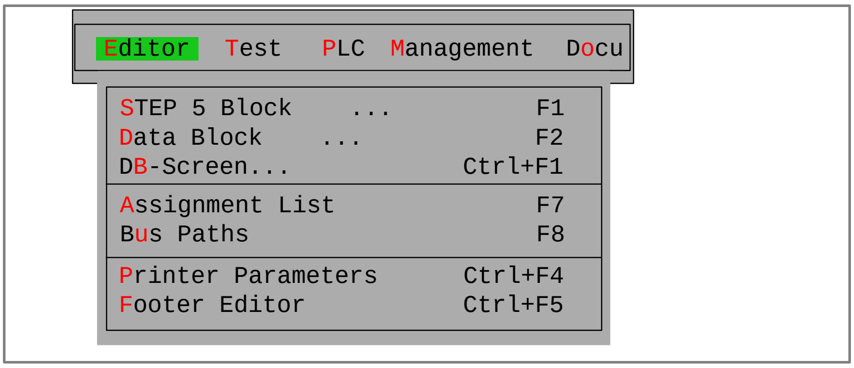
Part 2:
Editing with STEP 5

Common Functions in STL, LAD, CSF

5

Overview

This section describes all the functions that you can use in the three types of representation when editing.



Chapter Overview

5.1 Selecting an Editor

Overview

To edit

- STEP 5 blocks in the methods of representation LAD, CSF or STL
- comment blocks
- documentation blocks and
- plant comments

you must select an editor . The method of representation depends on the project setting (see **File ▸ Project ▸ Set F4**, Section *4.1.1*) but this can be changed when editing in the output mode using a function key.

Editor
`STEP5 Block`

Select the menu command **Editor ▸ STEP 5 Block**. The dialog box as shown in Figure 5-1 is displayed.

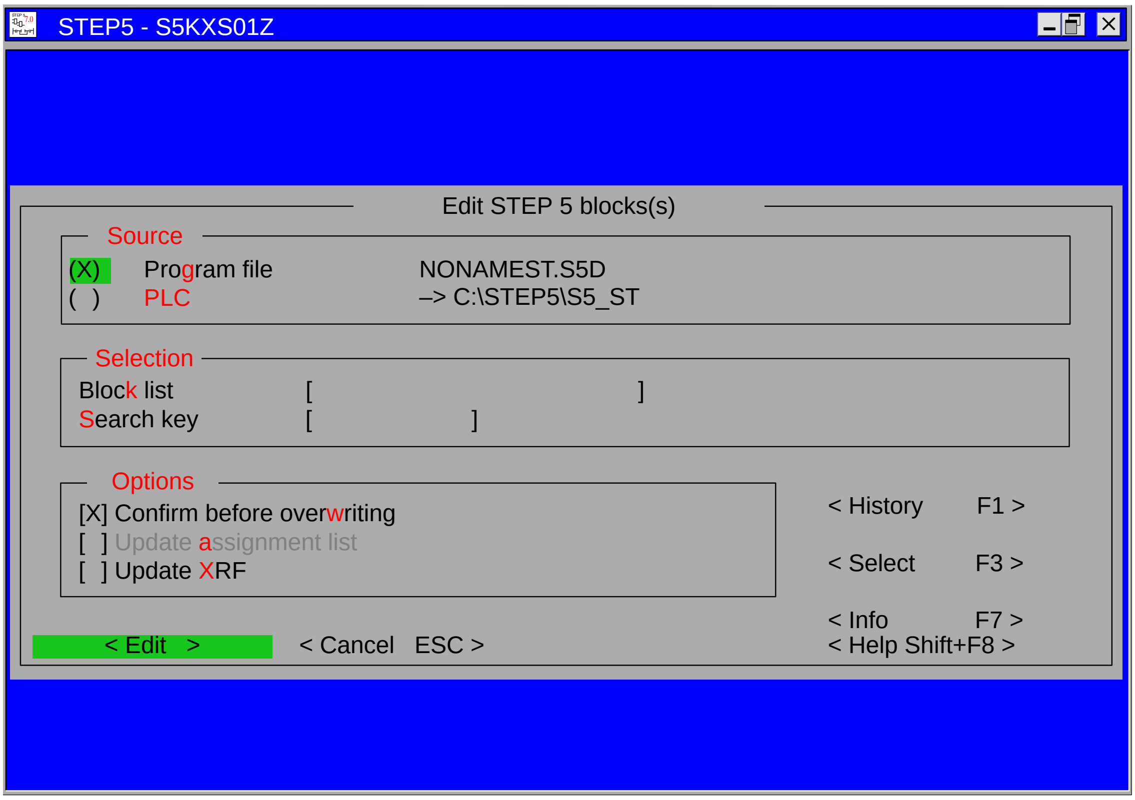


Figure 5-1 Edit STEP 5 Block(s) Dialog Box

Select your source and the options you require. The meaning of the fields is as follows:

Source

[x] `Program file`

Specifies where the file is stored or will be stored. This entry is made in the project settings (see Section *4.1.1*)

[x] `PLC`

Specifies the PLC on which the block is stored or will be stored. This entry is made in the project settings and is only possible in the online mode (see Section *4.1.1*)

Selection
Block list

`Block list [              ]`

In this input field, you can make the following entries depending on the dialog box:

- Single block
- Block list
- Block range
- Block type
- DOC block

If you press the ***F7*** key or click the ***< Info   F7>*** button, STEP 5 displays a list of the currently possible selections.

If you want to edit an existing block or display the currently permitted block types, press the ***F3*** key or select the ***< Select   F3 >*** button.

You can specify the block in absolute or symbolic form, a mixture is also possible.

Single block

Single block, specified in absolute or symbolic form.

```
[PB100                                ]
[DX 14                                ]
[OB 10                                ]
[FCX 231                              ]
[–Plant1                              ]
```

Block list

A list with a maximum of six single blocks. The blocks are separated by commas. If the comma follows a symbolic name, the comma must be preceded immediately by “\” to delineate the symbol.

```
[PB100 , PB123                        ]
[–Plant1\, –Plant2                    ]
[–Plant1\, FB45, –Plant2\,–Control    ]
```

Block range

A block range indicated by two blocks. The blocks are separated by a dash. If the dash follows a symbolic name, the dash must be preceded immediately by “\” to delineate the symbol. Both blocks must be of the same type and the first block number must be lower than the second.

```
[PB100 – PB123                        ]
[ –Plant1\- –Plant2                   ]
[–Plant1\- FB45                       ]
```

Block type

All blocks of one type.

```
[PB          ]all program blocks
[A           ]all blocks
[OC          ]all OB comments
[DB          ]all data blocks
[#           ]all DOC blocks
```

DOC block

A block preceded by the # character

```
[#MOT_P                          ]
[#DBDO.003                       ]
[OC#OBDO.024                     ]
```

Selection
Search key

`Search key[            ]`

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 or symbolic form.

Then enter an operand as the search key in absolute or symbolic form. Press the ***F7*** key or click ***<Info F7>*** to display the permitted search keys. The key is searched for in all the specified blocks.

Note

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.

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.

When a term is found, the segment is displayed in the output mode. If the first block entered does not exist, then after entering the parameters and options, the first segment (empty) of this block is displayed in the edit mode. After you exit the block, the program continues to search through the specified blocks. If a non-existent block is specified after the first block, the missing block is skipped.

You can continue the search for the key as follows:

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

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*** = *Enter* in the Edit mode and confirm the message E*nter modified segment?* with ***Yes***. The editor changes to the output mode.
2. Now press ***F7*** = *Enter* 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.

Options

[x] `Confirm before overwriting (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

[x] `Confirm before overwriting (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.

[x] `Update assignment list (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 save your input.

[x] `Update assignment list (no)`

The assignment list is not updated. You can, however, update or create the assignment list file later using the function → *INI > SEQ*

[x] `Update XRF (yes)`

The cross reference list (file *XR.INI) is updated when a block is modified.

[x] `Update XREF (no)`

The cross reference list is not updated. You can, however, update or create the cross reference list later using the function **Management** ▸ ***Make XRF*** (see Section 17.1)

5.2 Assignment of the Function Keys in the Output Mode

Overview

The following description of the keys provides you with an overview of the tools and functions available to support editing regardless of the type of representation.

F Addresses	F Lib No	F Symb. SYM	F	F –> LAD	F **Seg Com**	F Save	F Help
1 Disp Symb	2 Reference	3 Search	4	5 Seg Fct	6 Edit	7 Enter	8 Cancel

Table 5-1 Function Keys in the Output Mode

Function Key	Explanation
F1 *= Disp Symb*	Edit symbolic operands directly in the block.
F2 *= Reference*	Create, display references (cross references), change block.
F3 *= Search*	Search for single operands.
F4	–
F5 *= Seg Fct*	Page, copy, mark, insert, append and delete segments.
F6 *= Edit*	Change to the edit mode, also possible with the ***CORR*** key.
F7 *= Enter*	Save the block if it has been changed, or return to the main menu.
F8 *= Cancel*	Return to the main menu. Any changes to a block are discarded.
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/Line Com/Symb 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.

5.2.1 Inputting the Library Number (*SHIFT F2*)

Overview

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

Follow the steps outlined below:

1. Press ***SHIFT F2*** *= Lib No*
   the cursor is located in the displayed LIB field.
2. Type in the required LIB number 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 want to enter a number, exit the field with the ***ESC*** key.

5.2.2 Method of Representation *(**SHIFT F5** = -> LAD)*

Overview

With this function you can switch over the method of representation without having to call up the project settings (see Section *4.1.1*).

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 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*.

5.3 Editing Comments

Overview

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

- Plant comments
- Statement comments (*Editing Statement Lists, Chapter 6*)
- Segment comments
- Segment titles
- Operand comments (*Editing Assignment Lists, Chapter 11*)

Comments for data blocks DB and DX can be found in *Editing Data Blocks (see Chapter 9)*.

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

5.3.1 Plant Comment

Overview

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 per block. 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, make sure that the second character of the name is not a colon.

Working with the Editor

To enter or modify plant comments, follow the steps outlined below:

1. Select the menu command **Editor ▸ STEP 5 Block**.
2. Type in the name of the documentation block preceded by the character **#** and enter your selection .

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

F1	F2	F3	F4	F5
Insert	Delete	Command		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 editing.
F5 *= Insert L*	Insert a line at the cursor position.
F6 *= Delete L*	Delete the line at the cursor position.

F 1 Insert

You can insert ASCII characters within a text. Follow the steps outlined below:

1. Press ***F1*** *= Insert.* (change to the insert mode)
2. Type in the required string.
3. Change to the overwrite mode by pressing ***F1*** *= Overwrite.*

The entry of the text in the insert mode is completed.

F 2 Delete

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.

F
3 Command

The text editor has 8 commands for fast text processing. 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 Insert key.

The PG executes the command.

Table 5-2 Text Commands

Command	Effect of the Command
JTT	(jump to the top). From any position, the cursor jumps to the start of the comment.
JTE	(jump to the end). From any position, the cursor jumps to the end of the comment.
ST1, ST2, ST3, ST4	(set tag 1 etc.). You can set a maximum of 4 tags within the text.
JT1, JT2, JT3, JT4	(jump to tag 1 etc.). From any position in the text, the cursor jumps to the specified tag.
F/xyzrst/	(find). The cursor jumps to the selected text **xyzrst**, otherwise STEP 5 displays the message *not found*.
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.
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.
DT1, DT2, DT3, DT4	(delete). You can delete tags in any order.

Note

The printer control character **$EJECT** triggers a form feed in a segment, block or plant comment.

$EJECT must be in upper case letters, otherwise STEP 5 does not recognize the command.

Example

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

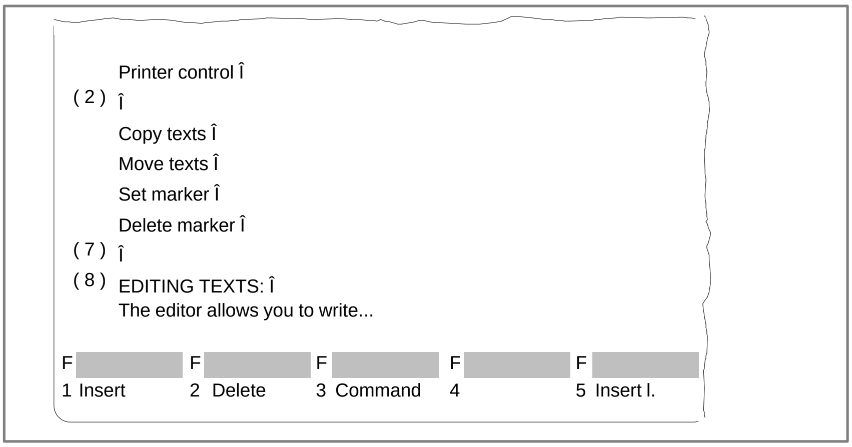


Figure 5-2 Printer Control

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 ***Insert*** key.

The cursor returns to the text.

Defining the End

1. 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.
2. Type in the characters *ST2* and press the ***Insert*** key.

The cursor returns to the text.

Copying a Block of Text

1. Position the cursor on the arrow in line (2) and press ***F3****.*
2. Type in the characters *CT1,T2* and press the ***Insert*** key. 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.

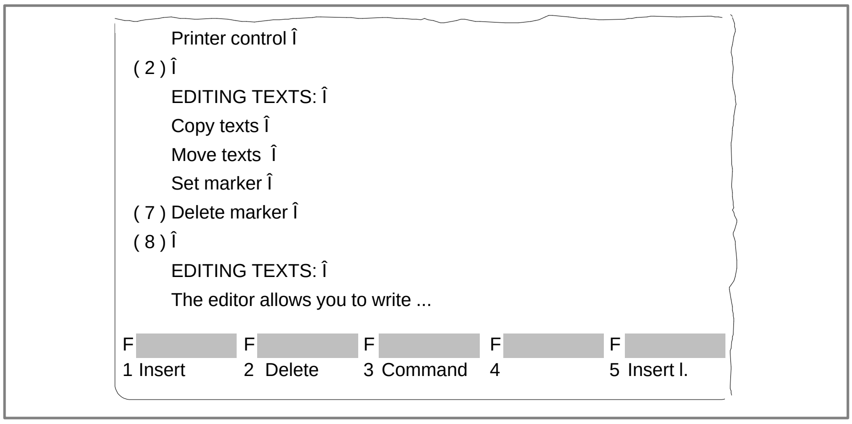


Figure 5-3 Printer Control

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 ***Insert*** key.

5.3.2 Segment Comment

Overview

Segment comments are texts with which you can write extra information about programs in segments or blocks. 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 Section 5.3.1.

- 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 PB13.
- 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

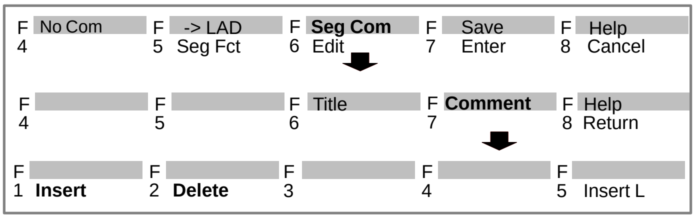
Note

You trigger a form feed with the printer control character **$EJECT.** This string must be written in upper case letters.

Ready to Start?

You have selected comments *[X]* in the project settings (Section *4.1.1*) 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

To enter or to modify a segment comment follow the steps below:

1. Select the menu command **Editor ▸ STEP 5 Block**.
2. Enter the block name
3. Press ***SHIFT F6*** *= Seg Com* and ***SHIFT F7*** *= Comment*

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. Press the ***F1*** key (Insert).

4. Edit the text using the alphanumeric keyboard.
5. Complete each line with the ***Return*** key.

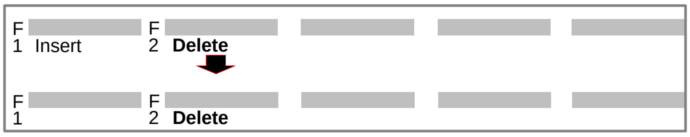
The end of the line is marked by a vertical arrow.

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 in the text where you want to insert characters.
2. Press ***F1*** *= Insert.*
3. Insert the required 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

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

Deleting a Line

1. Position the cursor in the line you want to delete.
2. Press ***F6*** or click the ***Delete L*** button.

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 save the block, STEP 5 also saves the segment comment.

Saving the Segment Comment

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

5.3.3 Segment Title

Overview

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 save 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 automatically assigns the comment block name as follows:

  OBn $\rightarrow$ OCn
  PBn $\rightarrow$ PCn
  SBn $\rightarrow$ SCn
  FBn $\rightarrow$ FCn
  FXn $\rightarrow$ FCXn

Ready to Start?

You have selected *comments: yes* in the *Settings* (*Section 4.1.1*).
If this is not the case, you can switch over by pressing ***SHIFT F4*** = *Line Com.* The segment in which you want to enter a title is open. STEP 5 is in the output or edit mode.

F4	F5	F6	F7	F8
No Com	–> LAD Seg Fct	**Seg Com** Edit	Save Enter	Help Cancel

F4	F5	F6	F7	F8
		Title	Comment	Help Return

Working with the Editor

To enter or to modify a segment title follow the steps below:

1. Select the menu command **Editor ▸ STEP 5 Block**.
2. Type in and enter the name of the documentation block.
3. 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.
4. Type in text or correct an existing text
5. Press the ***Return*** key.

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

5.3.4 Display Operand Comments

Overview

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 project settings and *Symbols* and *Display: symbolic* were 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 project 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 project settings

F	**No Com**	F	–> LAD	F	Seg Com	F	Save	F	Help
4		5	Seg Fct	6	Edit	7	Enter	8	Cancel

5.4 Appending, Inserting, Transferring, Deleting a Segment

Overview

F4 **No Com**	F5 –> LAD **Seg Fct**	F6 Seg com Edit	F7 Save Enter	F8 Help Cancel

F4 **Delete File**	F5 **Insert**	F6 **Append**	F7	F8 Help Return

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 (see *Appendix, Keyboard*).

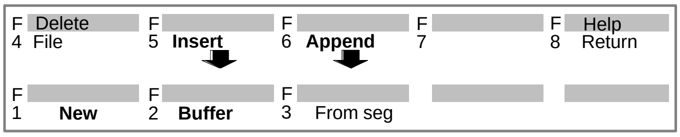
Segment editing function	Function keys	Key in numeric pad
Save segment temporarily	***F4*** *= File*	–
Insert before current segment	***F5*** *= Insert*	Insert segment
Append after current segment	***F6*** *= Append*	Segment end
Delete segment	***SHIFT F4*** *= Delete*	Delete segment
Page to existing segment	***F1*** *= –1* ***F1*** *= +1*	– +

5.4.1 Appending or Inserting a New Segment

Overview

Follow the steps outlined below:

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

5.4.2 Copying a Segment

Overview

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. You copy in the output mode.

Copying a Segment in the Same Block

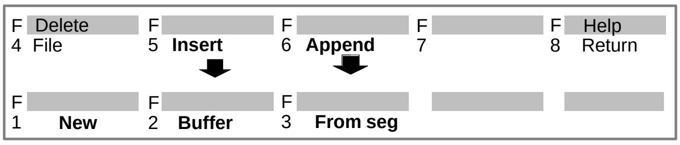
Note

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).

Copying a Segment

Follow the steps outlined below:

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

Follow the steps outlined below:

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

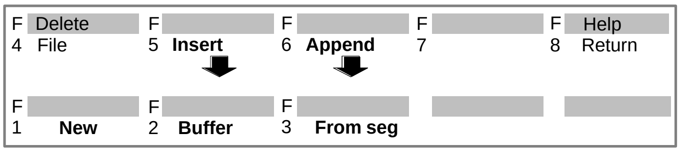
Filing (copying) the Segment

3. Press ***F4*** *= File.*
   The segment is temporarily stored.
4. Press ***F8*** *= Return.*

   Returns you to the block editor in the output mode.
5. Save any changes with SHIFT + F7 or F7=Enter. If you have made no changes, exit the block with the ***ESC*** *= Cancel* key.

Inserting the Segment

6. Change to the destination block with ***F2*** Reference and ***F4*** Dest Blk.
7. Press ***F5*** *= Seg Fct.*



8. Press ***F5*** *= Insert* again, if you want to insert before the current segment o*r* ***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.

5.4.3 Deleting a Segment

Overview

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

Follow the steps outlined below:

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.

5.4.4 Transferring/Moving a Segment

Overview

You can move a segment within the same block or transfer it to a different block in the same program file. This function is a combination of *copying a segment* (Section 5.4.2) and *deleting a segment (*Section 5.4.3*)*. After the transfer you must update the cross reference list 〈*Section 17.1*).

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.*

5.5 Creating, Displaying Cross References, Block Change

Overview

The cross references of all blocks in a program file are stored in a special program file *XR.INI. You can access this data in the editor window using the function ***F2*** = *Reference (see Section 5.2, Output Mode)*.

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 save a block,
- as an alternative, you can use the management function *Make XRF*.

5.5.1 Working with the Function *Make XRF*

Overview

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

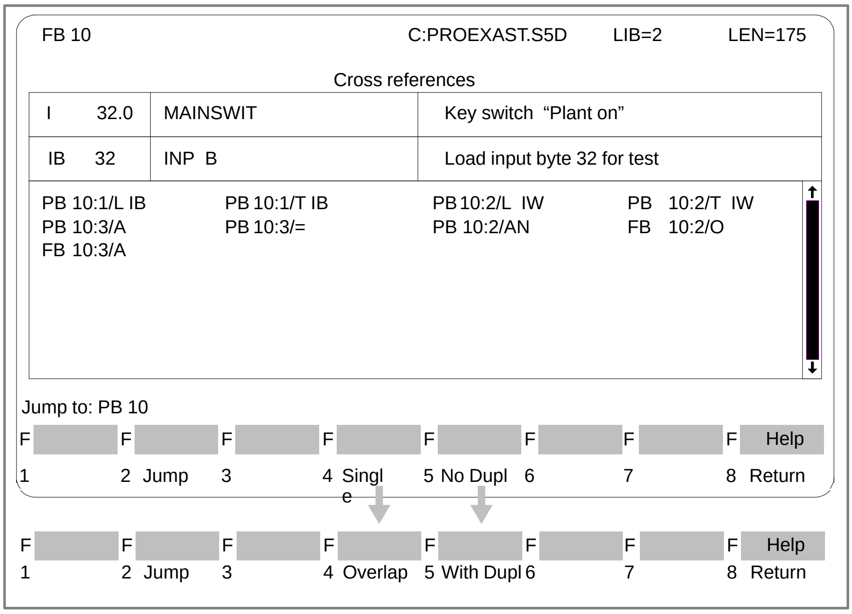
When creating the XRF within the editor, there is less memory available than for generating an XRF starting directly from the menu. This means that with large program files, data must be written to temporary files earlier. This slows down the creation of the XRF.

5.5.2 Display Cross References (Function *Display XRF*)

Overview

Follow the steps outlined below:

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:
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**= With Dupl/No Dupl:*
   With Dupl: if an operand occurs with the same operator more than once in a segment, it is displayed as often as it occurs.
   No Dupl: 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.

Note

Commands which are marked with # are commands with editing functions (B MW... or B DW...). The command which is actually executed during run-time is in this case unknown.

5.5.3 Changing Blocks

Jumping to a Block

Follow the steps below:

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 followed by* ***F5*** *= Orig Blk.*

Changing Blocks

Follow the steps below:

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.

5.6 Searching for Operands, Segments and Addresses

Overview

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, it 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
- Segments
- Addresses

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.

Searching for a Segment

1. Type in the segment as a decimal number.

Searching for an Address

1. Type in the address as a hexadecimal number. The last character of the number must be 'H'. In LAD/CSF, only the segment for this address is found. In STL, the cursor is positioned exactly on the address. If the address is too high, the end of the block is displayed as the result of the search.

5.7 Editing Symbolic Operands in the Block

Overview

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 assignment list* in the *Edit STEP 5 block(s)* 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* in the *project settings* (*Section 4.1.1*). If this is not the case, you can switch over with ***SHIFT F3***.

STEP 5 is in the Output mode.

How to Edit Symbolic Operands

Follow the steps below:

1. Press ***F1*** *= Disp Symb.*
   A list containing the operands is displayed on the screen.
2. Select the operand with the 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	FLAG 25	FLAG 25

4. Enter the symbolic name in upper and lower case characters.
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 in upper and lower case characters.
7. Complete the edited line by pressing ***F2*** *= Insert.*
8. To complete editing, press ***F8*** *= Return* or the ***Insert*** key.

Note

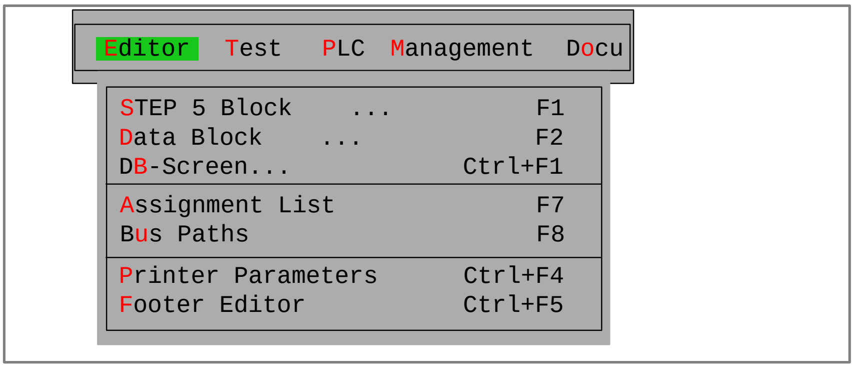
Symbolic names should begin without a hyphen. Do not use umlauts (ä, ü, ö).

Editing Statement Lists (STL)

6

Overview

This section describes the functions you can use in the STL type of representation.



Chapter Overview

Statements

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 operation and the operand as follows:

Example

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

You can write a maximum of 255 words per segment.

Note

For a detailed example of editing a Statement List, refer to Chapter 24 **Practical Application of STEP 5**.

6.1 General Aspects of Working with the STL Editor

Ready to Start?

Before you start editing, check the project settings with the menu command **File ▸ Project ▸ Set F4.** Make sure that the entries for the program file, symbols file, mode, type of representation and comments are correct.

Statements are always entered in the *Edit mode*. If you open a new block, STEP 5 is in the *Edit mode*, if you open an existing block, STEP 5 is in the output mode. In this case, you can change to the Edit mode with ***F6*** = *Edit*.

Starting the Editor

Editor
`STEP 5 Block F1`

Select the menu command **Editor ▸ STEP5 Block.** The *Edit STEP 5 block(s)* dialog box is displayed.

Once you have named your block, it is advisable to select the options *Update XRF* and *Update assignment list* if you are working with symbolic operands.

After confirming your entry with ***Edit***, the STL editor is started.

Screen Layout

A screen with a working field and a function key bar is displayed. Press **SHIFT F8 = Help** to display explanations of the function keys.

Typing in Statements

When typing in a statement, you do not have to keep to the strict format, in other words, STEP 5 enters the blanks automatically after you have entered the line. Complete each line with the ***Return*** key.

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.

Saving the Block

Press the ***Insert*** key. STEP 5 switches to the output mode. Press the ***Insert*** key again.

6.2 Simple Editing Functions

6.2.1 Displaying Addresses

Overview

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.

How to Display Addresses

Follow the steps below:

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

   STEP 5 displays the relative addresses in words.
2. Set the STL addresses to WORD or Byte (see *Section 4.1.1)*
3. Press ***SHIFT F1*** *= Addresses.* The display with addresses disappears and 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 or bytes depending on the PLC. If you press ***SHIFT F1*** *= Addresses* a second time, the address information is cleared from the screen. The addresses are displayed in hexadecimal format !

6.2.2 Statement Comment

Overview

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 save 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 *[X] with comments* in the *project settings* (*Section 4.1.1*). 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

Follow the steps below:

1. Position the cursor on the required statement.
2. Move the cursor to the right to the comment field *(**SHIFT + 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.

6.2.3 Saving the Comment

Overview

The first time you save 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 comment in file?.`

Enter the comment with the ***Insert*** key or discard it with ***ESC*** = *Cancel*.

6.3 Function Block

Overview

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).

6.3.1 Editing a Function Block

Overview

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 in graphic form 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 (**Editor ▸ STEP 5 Block** with the option [x] FBs with name.)

```
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

F Addresses  F Lib No     F Synb.SYM  F No com  F —>LAD   F Seg Com  F Save   F Help
1 Disp Symb  2 Reference  3 Search    4         5 Seg Fct 6 Edit     7 Enter  8 Cancel
```

Figure 6-1 Example of a Function Block (FB 200)

Field Names

Table 6-1 Meaning of the Fields

Field	Meaning
`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.

Table 6-1 Meaning of the Fields, continued

Field	Meaning
`I/O/D/B/T/C`	The type of formal operand: I Input parameter O 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`	The type of formal operand: 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

STEP 5 is in the edit mode (STL). Segment 1 is open. If you are using symbolic operands, a symbols file must exist and *symbols: yes* must be set in the *project settings*. With a new function block, follow the steps below:

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  :
```

Figure 6-2 Parameter List for Formal Operands

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.

Modifyng a Function Block

When you call the function block, the actual operands are assigned to the formal operands. STEP 5 is in the edit mode. The function block to be called is in the program file.

How to Assign Parameters

Follow the steps below to modify a block:

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.

Note

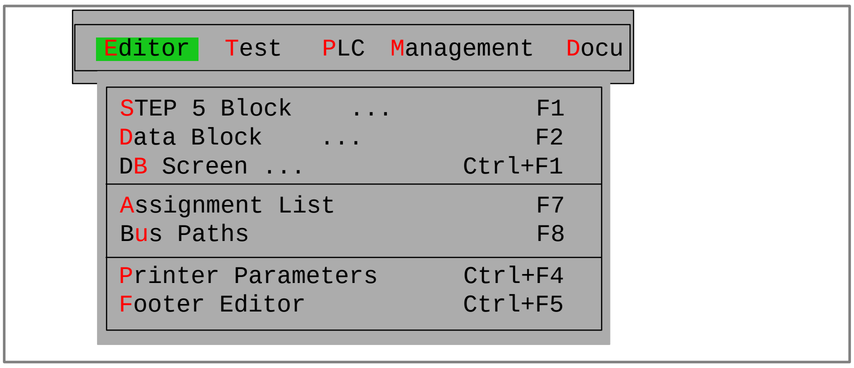
To insert segments, use the function keys ***F5*** or ***F6***. Using the STL command *** can lead to undesired program structures.

Editing Ladder Diagrams (LAD)

7

Overview

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 blocks OB
- program blocks PB
- sequence blocks SB
- function blocks FB
- extended function blocks 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.

Chapter Overview

7.1 General Aspects of Working with the LAD Editor

Ready to Start?

Before you start editing, check the project settings with the menu command **File ▸ Project ▸ Set F4.** Make sure that the entries for the program file, symbols file, mode, type of representation and comments are correct.

When editing existing blocks, you can change the type of representation with ***SHIFT F5*** = LAD (press once or twice).

Starting the Editor

Editor
STEP5 Block

Select the menu command **Editor ▸ STEP5 Block.** The *Edit STEP 5 block(s)* dialog box is displayed.

Once you have named your block, it is advisable to select the options *Update XRF* and *Update assignment list* if you are working with symbolic operands.

After confirming your entry with ***Edit***, the LAD editor is started.

Screen Layout

A screen with a working field and a function key bar with symbols for entering contacts and editing LAD segments is displayed.

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 content of the screen can be scrolled 2.5 times up or down. Press ***SHIFT F8*** *= Help* to obtain an explanation of the function keys on the screen.

Making Input

The editing field is divided into lines and columns in which you enter rungs, branches, contacts, outputs and function elements using function keys or the mouse.

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.

Ladder Diagram Representation

Figure 7-1 shows an example of a segment in the LAD representation.

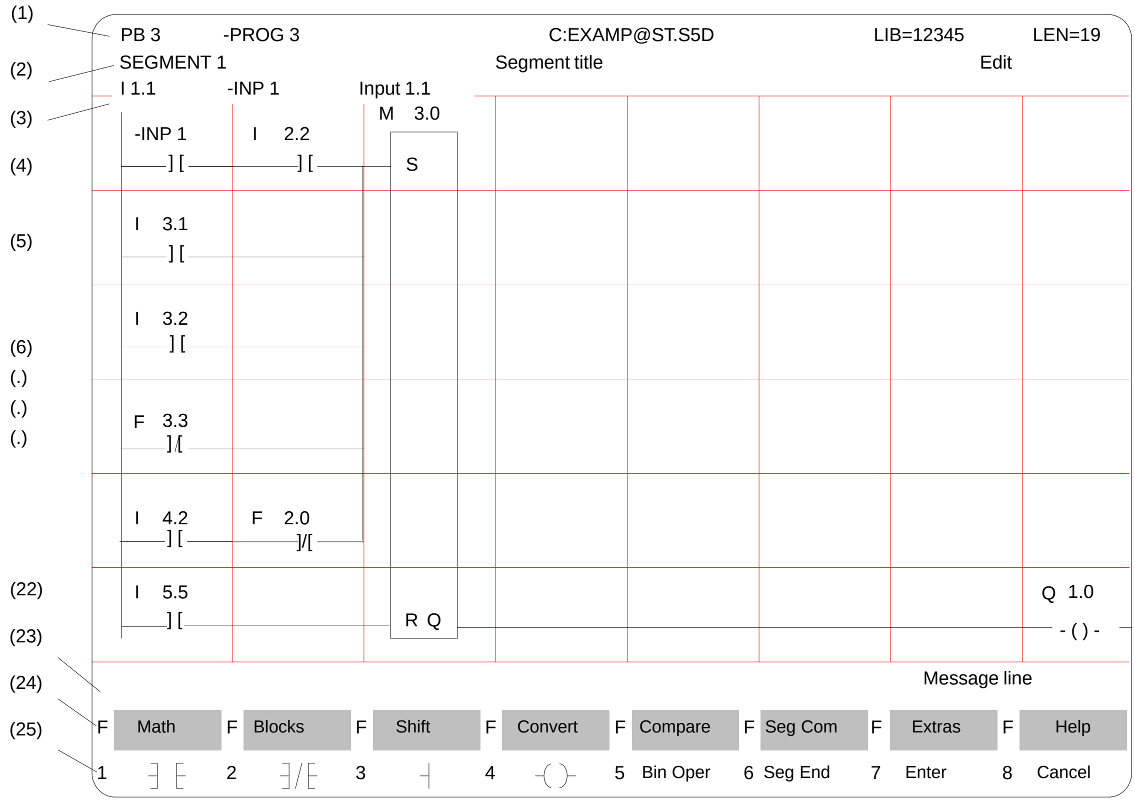


Figure 7-1 A Segment in Ladder Diagram Representation (Example)

Screen Lines

The screen lines have the following meaning:

Table 7-1 Explanation of the Screen Lines

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

7.2 Simple Editing Functions

Logic Operations

After you have selected the editor, 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 (*Table* 7-2) 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 7-2 Logic Operations in LAD (Ladder Diagram)

Operation	Function Keys	Explanation
—| |—	***F1***	NO contact
—|/|—	***F2***	NC contact
⊣	***F3***	Branch, close branch
—( )—	***F4***	Output
Bin Oper	***F5***	Call complex functions
—(#)—	***F5+F4***	Connector
—(/)—	***F5+F5***	Negated connector
——	[→] ***(Cursor right)***	Empty element

Note

In LAD it is only possible to use assignments (=) for outputs. Programs produced in STL with the outputs Set (S) and Reset (R) are issued with S and R in LAD.

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.

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 *assignment list* if you have not already selected this function in the job box.

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 7-2*).

Further parallel rungs are connected to this continuous rung. A parallel rung 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 rung 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*** = *Exp Hor* or the ***expand (horizontal)*** *key.*

Series

- Position the long cursor on the contact after the insertion point and press ***SHIFT F7*** = *Extras* and ***F6*** = *Exp Hor.*

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*** = *Bin Oper* + ***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.

- 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 rung.

Redisplaying a Segment

If you have a segment that has become an awkward shape (for example due to repeated expanding), you can press ***SHIFT F7*** *= Extras* and ***F2*** *= New Disp* and redisplay the segment even if it does not yet have all the parameter values. The display is then refreshed and the presentation is optimized.

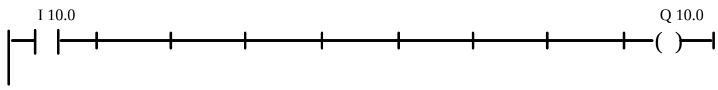
Note

You can only exit a segment or block when all the names and parameters have been input correctly.

7.3 Examples of Editing Logic Operations

Initial Situation

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:



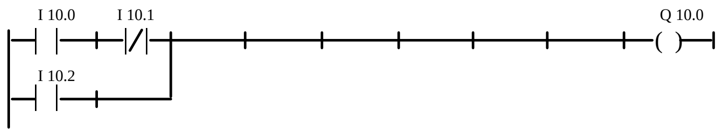
Series and Parallel Contact

Series contact:

1. Position the cursor in the second column of the displayed rung and press ***F2*** *= NC contact* type in **I 10.2** and press the ***Return*** key.

Parallel contact:

2. 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.



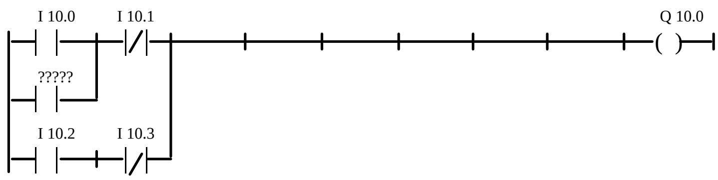
Implicit Expanding

Inserting an NO contact in a further parallel branch:

3. 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

Contact **I 10.3** is generated by positioning the cursor on the empty element and pressing ***F2*** *= NC contact.*

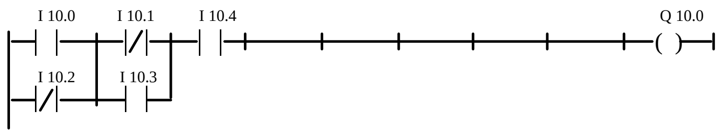


Bridge Circuit

You can obtain the bridge circuit below as follows:

4. 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.*

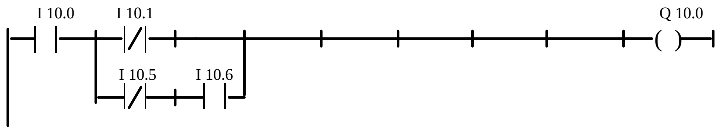
5. Creating the parallel branch: position the cursor below contact **I 10.0**, press ***F2*** *= NC contact* and ***F3*** *= Close branch* and position the cursor in the second column of the parallel branch, press ***F1*** *= NO contact* and ***F3*** *= Close branch.*



Opening a Branch after a Contact

The following segment shows a parallel path opened after the first contact.

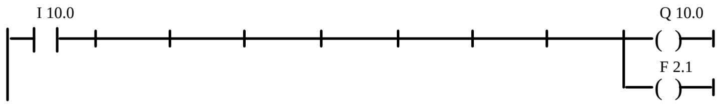
6. In the upper rung, position the cursor on the second column and press ***F2** = NC contact* for **I 10.1**.
7. Creating the parallel branch: position the cursor below contact **I 10.1**, press ***F2** = NC contact, **F1** = NO contact* and ***F3** = Close branch.*



Assignment

Connecting an output or an assignment:

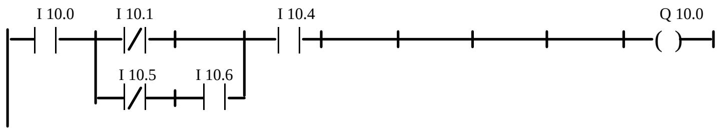
8. Position the long cursor under output **Q 10.0** and press ***F4** = Output.*



Using Connectors

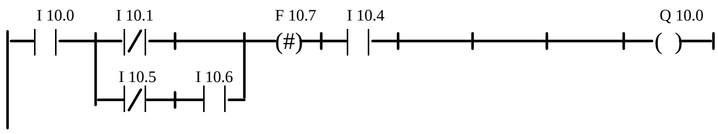
Connectors and negated connectors (*Table 7-2*) are intermediate flags in binary logic operations. They store the RLO formed up to that point. A connector is input in LAD in the same way as a contact. If it is located after the last contact of the rung it is represented as an output after the rung is 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:

9. Position the cursor on the contact below **I 10.4** and press ***DEL***.
10. Now position the cursor on the empty element and press ***F5** = Bin Oper* and ***F4** = Connector* to create a connector which you can then label **F 10.7**. Following this, insert contact ***I 10.4*** again.



7.4 Complex Functions

Overview

In the editing mode, the following functions can be called with ***SHIFT*** and a function key or ***F5*** = *Bin Oper*:

Table 7-3 Complex Functions in LAD

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 (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"
Bin Oper CD, CU	***F5*** and	 ***F1, F2***	(9) **Counter operations:** counter value incremented, decremented by 1
Bin Oper 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
#	***F6*** and	***F4***	**Connector**

Rules for Representation

The following rules apply to the non-elementary operations listed in Table 7-3 in LAD:

1. All operations (1) to (10) in Table 7-3 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 function keys 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.

7.4.1 Arithmetic Operations

Overview

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)

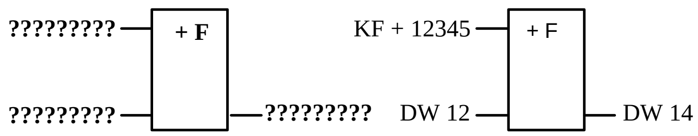
Operand types : KF, DW, IW...

Example

Editing an ADD operation for two fixed point numbers:

1. Press ********* or ***F6*** *=Seg End* 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.

7.4.2 Block Calls

Overview

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 function keys. In existing segments, you can insert and append calls with/without implicit expanding of the rung.

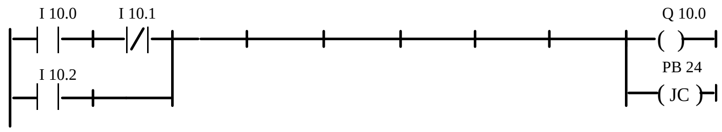
Note

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:

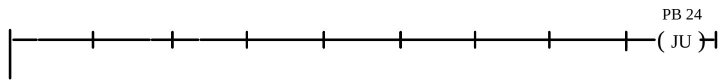
1. Position the cursor below the output symbol and press ***SHIFT F2*** *= Blocks* and ***SHIFT F4*** *= JC ....*
2. Enter the destination block, in this case PB 24, in the input field above the call symbol and complete the entry with the ***Return*** key.



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.*

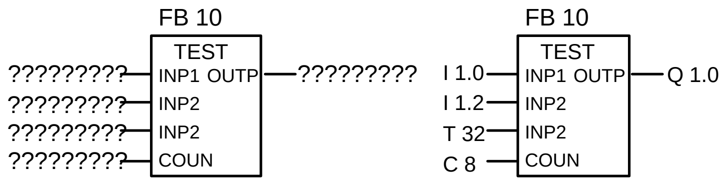
The editor displays the “roof” of the block with the cursor in the labelling field.

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.



7.4.3 Load and Transfer Operations

Overview

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 [?????].

7.4.4 Shift and Rotate Operations

Overview

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 function keys ***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*** *=Seg End* 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 possible to change parameter n by selecting the long box and positioning the cursor on the parameter with Shift –>.

7.4.5 Latching Operations

Overview

Using the latching functions, the RLO can be stored. You can specify how the latching function works after pressing ***F5*** = *Bin Oper* and then selecting either ***F6*** *priority set* or with ***F****7 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	0 with S/R element 1 with R/S element

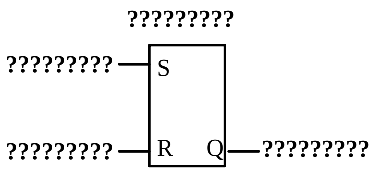
After pressing ***F5*** = *Bin Oper* and the required function key at the second key level, STEP 5 generates an undefined long box at the position of the long cursor in a LAD segment.

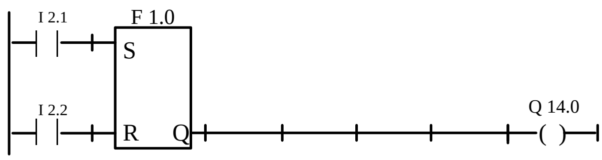
Example

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*** *= Bin Oper* 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*.*

7.4.6 Conversion Operations

Overview

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 [?????].

7.4.7 Comparator Operations

Overview

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

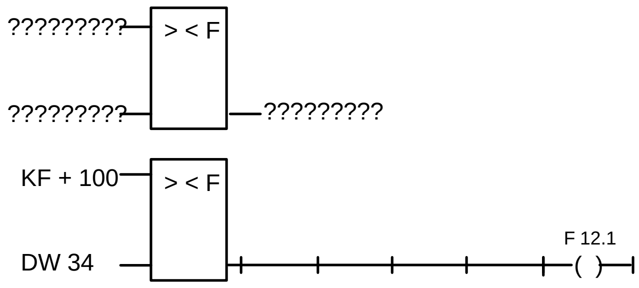
The type can only be modified directly once after activating the long box.
To change the type:
1. Position the long cursor on type
2. With ***Shift*** *+ cursor right,* position the small cursor on the type identifier
3. Change the type

Example

Operation to compare two fixed point numbers:

1. Open a new segment with ********* or ***F6*** *=Seg End* 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.

7.4.8 Digital Logic Operations

Overview

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:

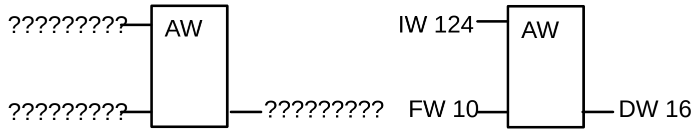
- load operand 1 (DW, IW, FW...),
- load operand 2 (DW, IW, FW...),
- combine the operands as words (AW, OW, XOW),
- transfer the result to operand (DW, IW, FW...).

Example

AND operation of two operands in words.

1. Open a segment with ********* or ***F6*** *=Seg End* 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.

7.4.9 Counter Operations

Overview

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*** = *Bin Oper* 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 $\rightarrow$ 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”.

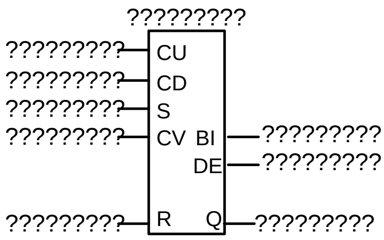
Counter operand: C 0 ... C 255
Range of values: 0 ... 999

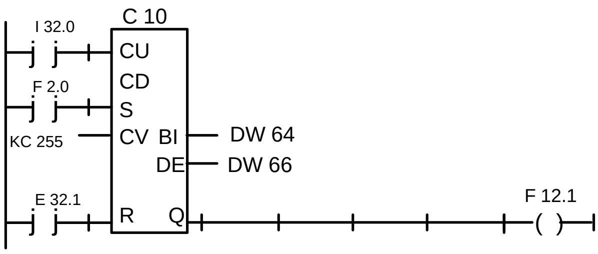
Example

Editing a counter function *count up*.

1. Open a segment with ****** or F6** =Seg End* and then press ***F5** = Bin Oper* 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.

7.4.10 Timer Operations

Overview

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*** = *Bin Oper* 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.

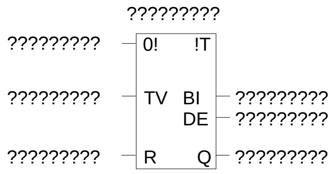
R	Reset input for the timer function. When this operand changes to 1, the timer and Q are set to 0.
BI	Current timer value, binary coded.
DE	Current timer value, BCD coded.
Q	Output indicating that the timer is running (Q = 1) or stopped or elapsed (Q = 0). Timer number: T 0 ... T 255

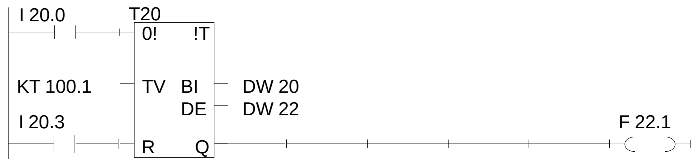
Example

Editing a timer function with OFF delay.

1. Open a segment with ****** or F6** =Seg End* and then press ***F5** = Bin Oper+* ***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.

Editing Control System Flowcharts (CSF) 8

Overview

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).

```
 Editor  Test   PLC  Management   Docu
  STEP 5 Block    ...          F1
  Data Block    ...            F2
  DB Screen ...            Ctrl+F1
  Assignment List              F7
  Bus Paths                    F8
  Printer Parameters       Ctrl+F4
  Footer Editor            Ctrl+F5
```

You can program in the Control System Flowchart representation in the following STEP 5 blocks:

- organization blocks OB
- program blocks PB
- sequence blocks SB
- function blocks FB
- extended function blocks 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.

Chapter Overview

8.1 General Aspects of Working with the CSF Editor

Ready to Start?

Before you start editing, check the project settings with the menu command **File ▸ Project ▸ Set F4.** Make sure that the entries for the program file, symbols file, mode, type of representation and comments are correct.

When editing existing blocks, you can change the type of representation with ***SHIFT F5*** = CSF (press once or twice).

Starting the Editor

Editor
`STEP5 Block`

Select the menu command **Editor ▸ STEP5 Block.** The *Edit STEP 5 block(s)* dialog box is displayed.

Once you have named your block, it is advisable to select the options *Update XRF a*nd *Update assignment list* if you are working with symbolic operands.

After confirming your entry with ***Edit***, the CSF editor is started.

Screen Layout

A working field appears on the screen and the function key menu with the symbols for entering functions and editing CSF segments.

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 screen can be scrolled a maximum of 2.5 times up or down. Press ***SHIFT F8*** *= Help* to obtain an explanation of the function keys on the screen.

Editing

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.

CSF Representation

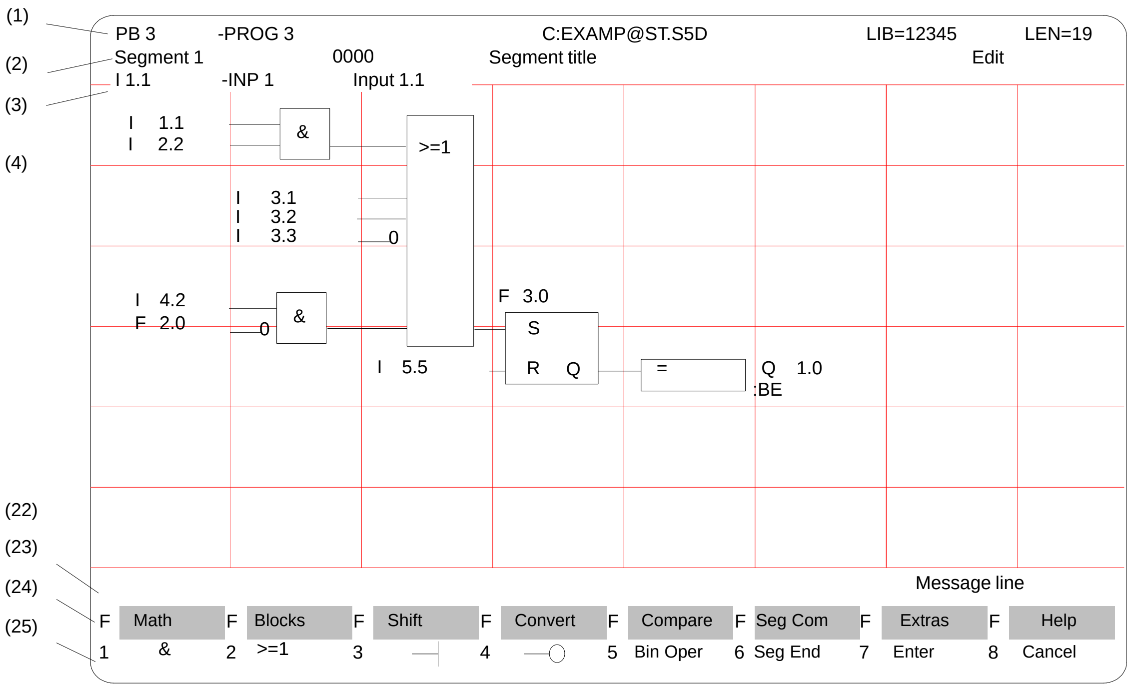


Figure 8-1 Segment in Control System Flowchart (Example)

Screen Lines The lines on the screen have the following meaning:

Table 8-1 Explanation of the Screen Lines

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

8.2 Simple Editing Functions

Logic Operations

After you have selected the Editor, 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 8-2*). 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 8-2 Logic operations in CSF (Control System Flowchart)

Operation	Function keys	Explanation
&	***F1***	AND operation
> = 1	***F2***	OR operation
—|	***F3***	Input
—○	***F4***	Negated input
Bin Oper	***F5***	Call complex functions
#	***F5*** and ***F4***	Connector
/	***F5*** and ***F5***	Negated, connector

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 *project 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.

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 *assignment list* if you have not already selected this function in the job box.

Note

You can only change the operand of a connector using the function “Delete” and “New Entry.”

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.

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).

8.2.1 Editor Functions: Modifying and Deleting

Overview

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):

1. You can append by positioning the long cursor below the lowest input of the long box.
2. 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**:

3. Place the cursor on the corresponding operand identifier and press ***F1*** = & or ***F2*** = >=1.

You **invert** an input

4. by positioning the cursor on the operand identifier and pressing ***F3*** = *Input* or ***F4*** = *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, see *Figure 8-2* (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, see *Figure 8-2* (B).
3. A function element with two operand inputs is removed. The remaining operand then occupies the free input of the next block, see *Figure 8-2* (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.

Example

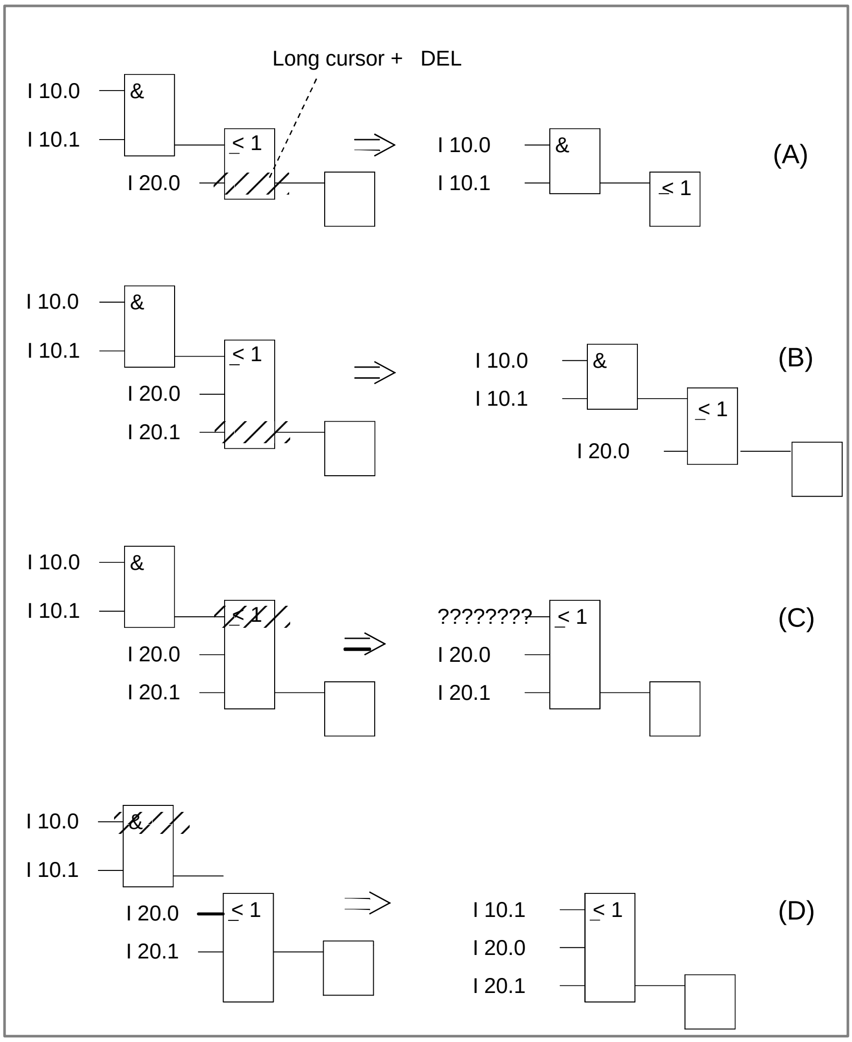


Figure 8-2 Deleting Operands and Functions (Example)

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 added to the bottom of the block (A).

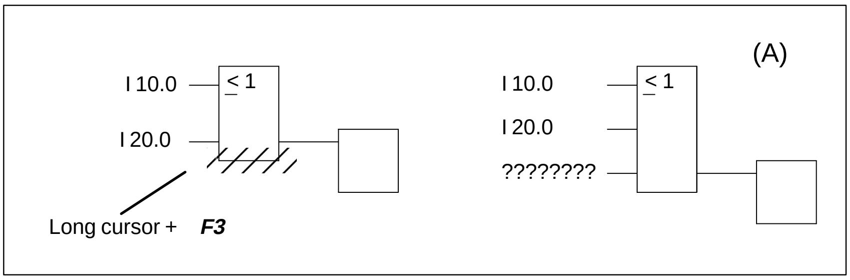


Figure 8-3 Appending Operands

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.

Horizontal and vertical expanding, i.e. in this case moving the segment to the right and down is performed implicitly.

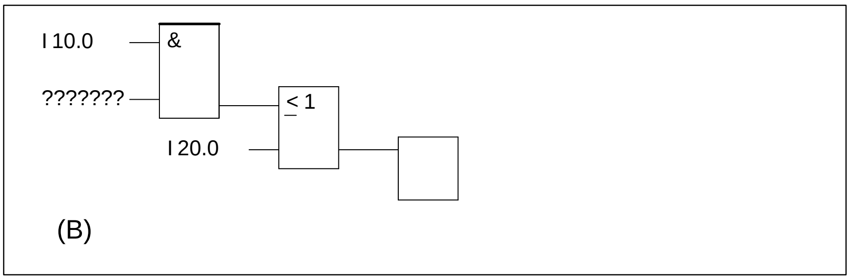


Figure 8-4 Appending a Function Block

Inserting Operands

Follow the steps outlined below:

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*** *= Exp Vert* and then ***F3*** *= Input.*

A non-connected operand is inserted in the block. After you have named the operand, you can invert the input with ***F4***.

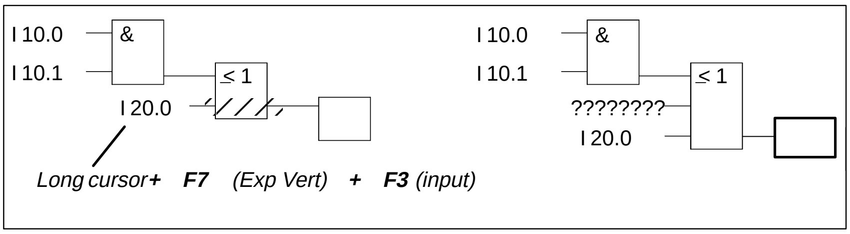


Figure 8-5 Inserting Operands

Inserting a Function Block

Follow the steps outlined below:

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*** *= Exp Hor* and select the required function, in this case ***F1*** *= &.*

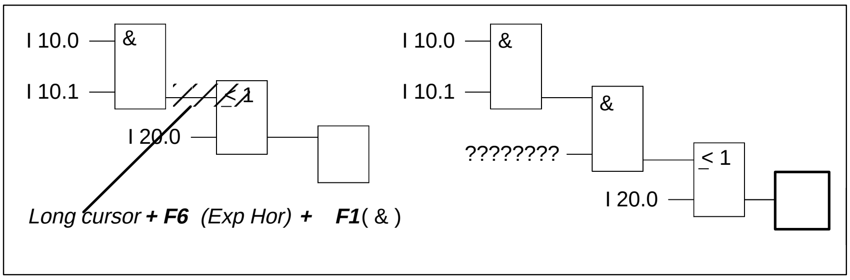


Figure 8-6 Inserting a Function Block

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 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. Name the connector, e.g. **F20.1** (A) and press the ***Return*** key.

Connector Stack

You obtain a connector stack by

2. Positioning the cursor on the connector and pressing ***F5*** = *Bin Oper* 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).

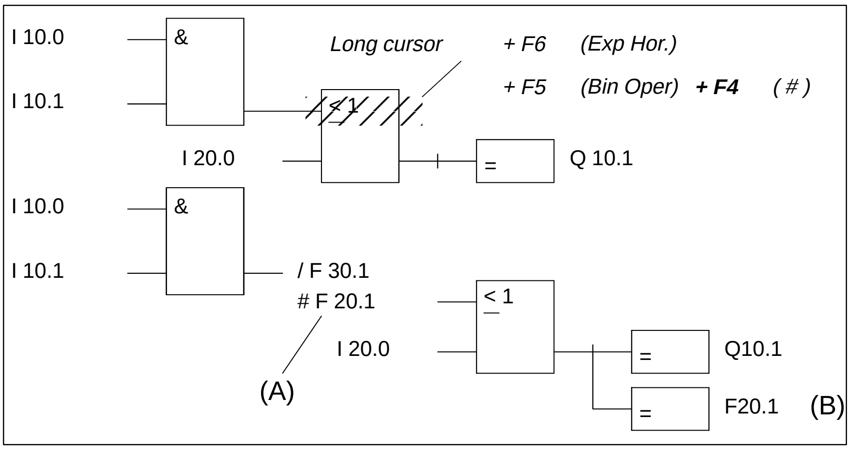


Figure 8-7 Editing connectors

You can delete a connector by positioning the cursor on the connector and pressing ***DEL***.

8.3 Complex Functions

Overview

In the editing mode, the following functions can be called with ***SHIFT*** and function keys or ***F5*** *= Bin Oper*.

Table 8-3 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
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*
Bin Oper CD, CU	***F5*** *and*	 ***F1, F2***	(9) **Counter operations:** Counter value incremented, decremented by 1
Bin Oper 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
#	***F6*** and	***F4***	**Connector**

Rules for Representation

In CSF, the following rules apply to the complex functions listed in Table 8-3:

1. All operations (1) to (10) in Table 8-3 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 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.

8.3.1 Arithmetic Operations

Overview

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*** *= Seg End* 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*.

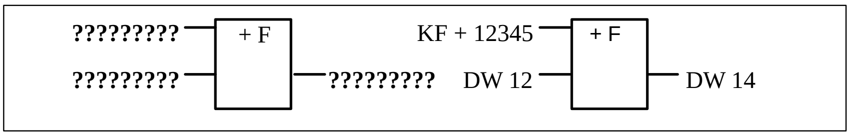


Figure 8-8 Editing an Add Operation

3. Confirm the operand format with the ***Return*** key.
4. Type in the 1st operand, here 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.

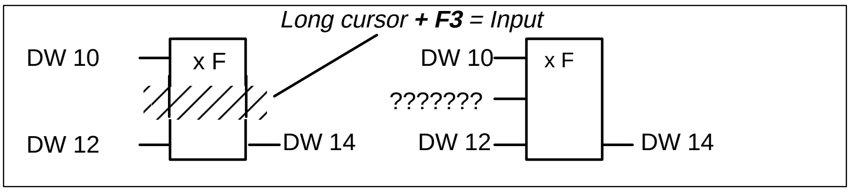


Figure 8-9 Inserting an Input

Appending an Input

Position the long cursor at the bottom of the function box and press ***F3*** *= Input* and name the input.

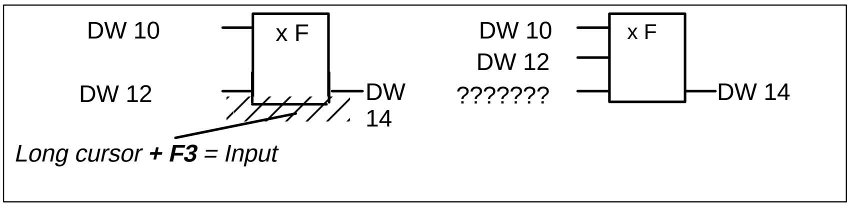


Figure 8-10 Appending an Input

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.

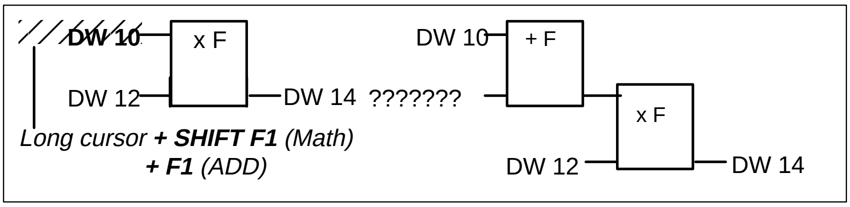


Figure 8-11 Inserting a Complex Function at an Input

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.

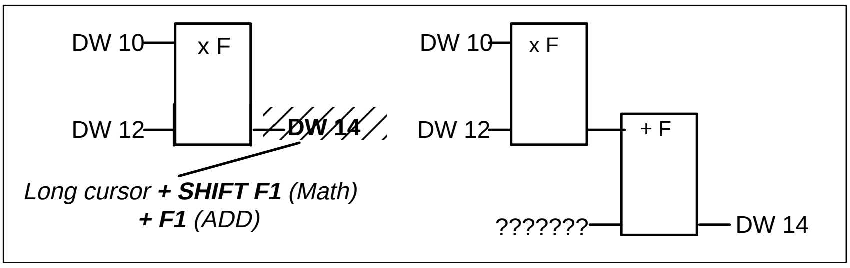


Figure 8-12 Appending a Complex Function at an Output

8.3.2 Block Calls

Overview

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 function keys.

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.

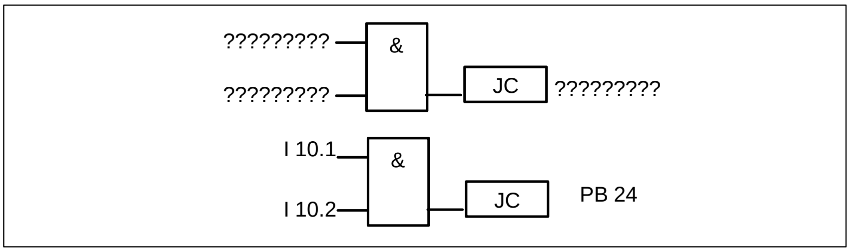


Figure 8-13 Conditional Program Block Call

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.

Figure 8-14 Unconditional program block call

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.

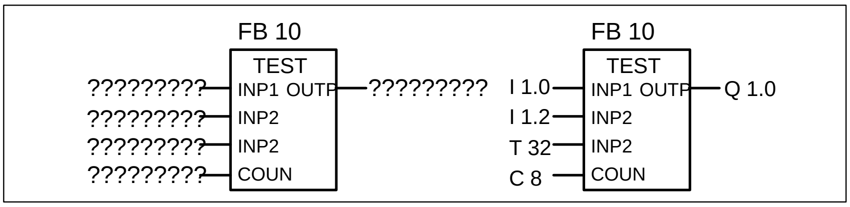


Figure 8-15 Unconditional FB Call

The segment then appears as shown on the right-hand side.

8.3.3 Loading and Transfer Operations

Overview

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 [?????].

8.3.4 Shift and Rotate Operations

Overview

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 function keys ***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*** = *Seg End* followed by ***SHIFT F3*** *=Shift.*
2. Select the required operation, in this case ***F2*** *= SRW.* STEP 5 displays the long box (left).

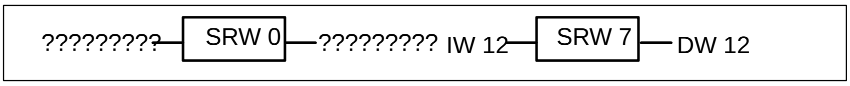


Figure 8-16 Shifting the Input Operator

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.

8.3.5 Latching Operations

Overview

Using the latching functions, the RLO can be stored statically outside the processor. You can specify how the latching function works after pressing ***F5*** = *Bin Oper* 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	0 with S/R flip flop 1 with R/S flip flop

After pressing ***F5*** = *Bin Oper* 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*** *= Seg End* and then ***F5*** *= Bin Oper* and ***F7*** *= S/R.*

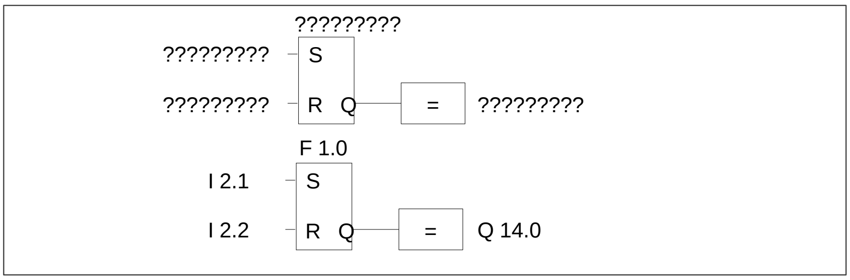


Figure 8-17 Editing a Latching Operation

2. Type in the operand ID for the memory location, in this case **F 1.0** and press the ***Return*** key.
3. *type in the input operands* **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.

8.3.6 Conversion Operations

Overview

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 [?????].

8.3.7 Comparator Operations

Overview

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.

Changing the type:
1. Position the long cursor on type
2. Position the small cursor on the type letter with ***Shift*** *+ Cursor right*
3. Change the type

Example

Operation to compare two fixed point numbers:

1. Open a new segment with ******* or ***F6*** *= Seg End* 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*.

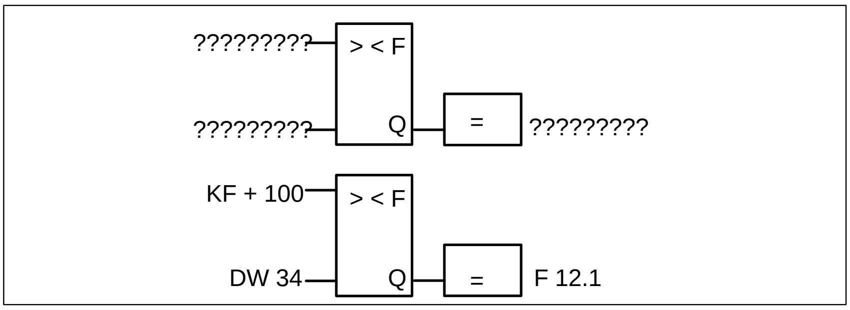


Figure 8-18 Editing Compare Operations

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.

8.3.8 Digital Logic Operations

Overview

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 on two operands in words.

1. Open a segment with ********* or ***F6*** *= Seg End* 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*.

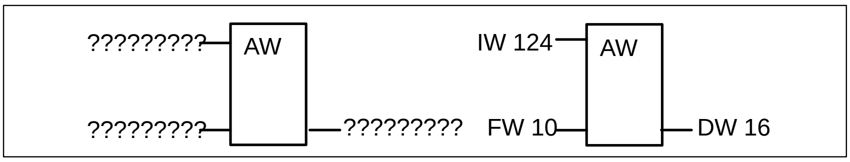


Figure 8-19 AND Operation

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.

8.3.9 Counter Operations

Overview

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*** = *Bin Oper* 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*** *= Seg End* and then press ***F5*** *= Bin Oper* and ***F2*** *= CU.* STEP 5 displays the long box with the undefined inputs/outputs.

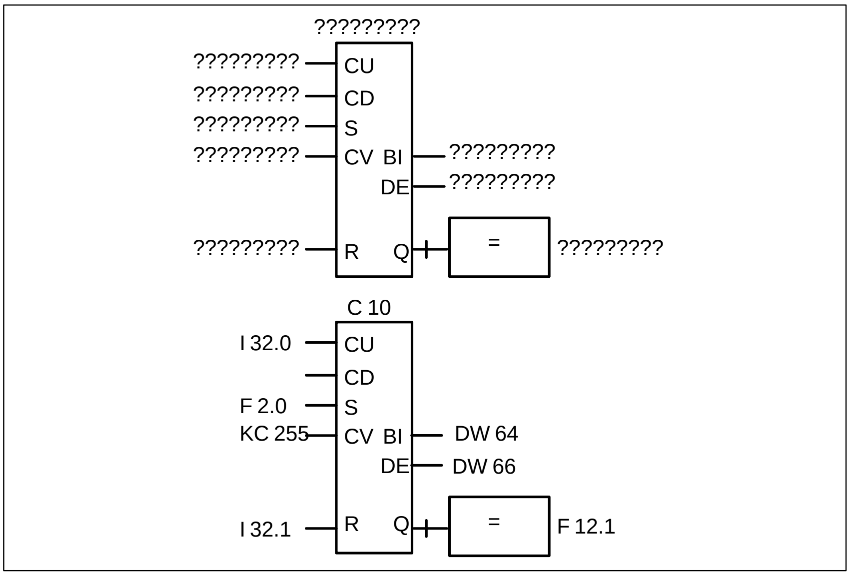


Figure 8-20 Editing a Counter Function

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.

8.3.10 Timer Operations

Overview

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*** = *Bin Oper* 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** = Seg End* and then press ***F5** = Bin Oper* + ***SHIFT F5** = SF.*

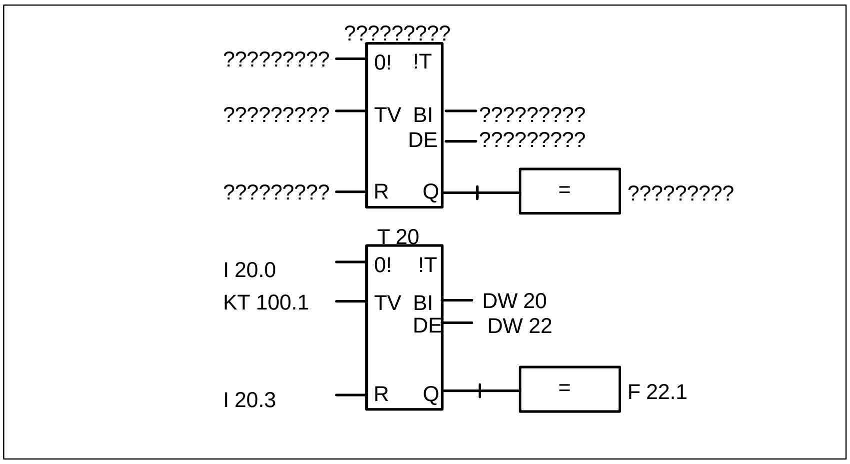


Figure 8-21 Editing a Timer Function with Off Delay

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.

9 Editing Data Blocks

Overview

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.

```
 Editor   Test   PLC  Management   Docu
 ---------------------------------------
 STEP 5 Block    ...          F1
 Data Block    ...            F2
 DB Screen ...           Ctrl+F1
 ---------------------------------
 Assignment List              F7
 Bus Paths                    F8
 ---------------------------------
 Printer Parameters      Ctrl+F4
 Footer Editor           Ctrl+F5
```

After introducing you to the basics of editing a data block, the individual functions of the editor are described separately.

Chapter Overview

9.1 Structure of a Data Block

Overview

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.

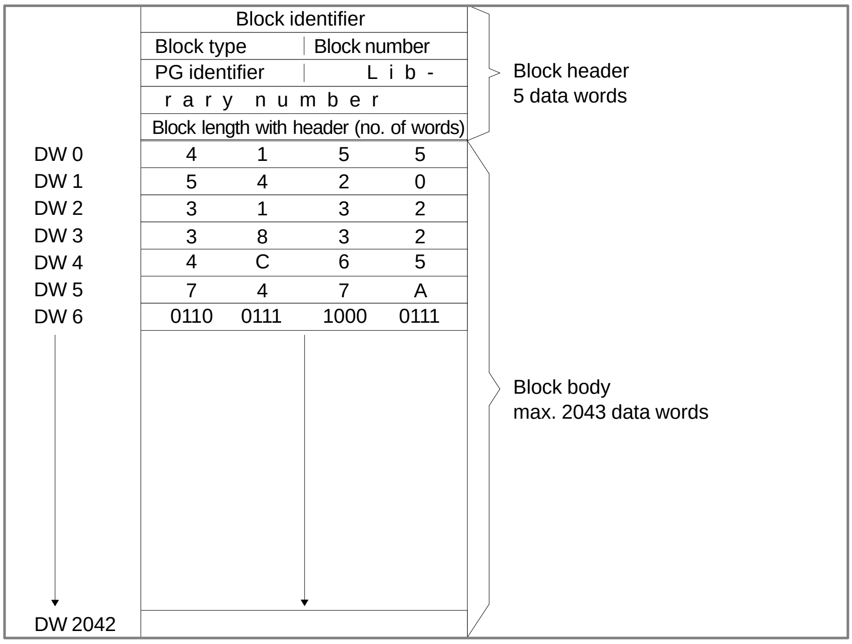


Figure 9-1 Structure of a Data Block

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.

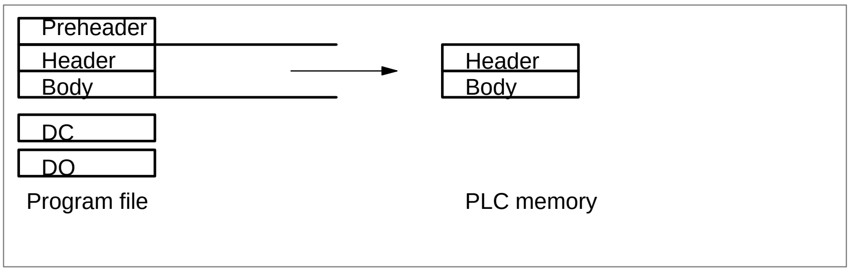


Figure 9-2 Block Preheader

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)
- Programmin device ID
- Library number (number between 0 and 99999)
- Block length (including the length of the block header)

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 data words. 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.

9.2 Editing Data Blocks

Calling the Editor

Editor
`Data Block`

Select the menu command **Editor ▸ Data Block...**. The *Edit data block(s)* dialog is displayed on the screen.

If you want to edit a data block, type in its name in absolute format (for example DB15) or its 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.

If you press ***SHIFT F8*** = *Help*, STEP 5 displays a list of possible inputs.

Click ***< Edit >*** to open the data block editor.

Screen Layout

Figure 9-3 shows the editing field of the DB editor with the function keys of the basic menu and a displayed data block.

You can press ***SHIFT F8*** = *Help* to obtain an explanation of the function keys on the screen.

Entries

The editing field is divided into lines and columns in which you enter data using the function keys menu or the mouse.

Saving a Block

Press ***F7*** = Enter or the ***Insert*** key.

Canceling the Function

Press ***ESC***.

If you interrupt the intended sequence with another operation, the PG displays the message: `First finish repetition factor!` The operation cannot be executed at this point because the editor is in the repetition mode and this must first be terminated.

Input Field

Figure 9-3 shows the editing field of the DB editor with the function keys for the basic menu and with a data block displayed.

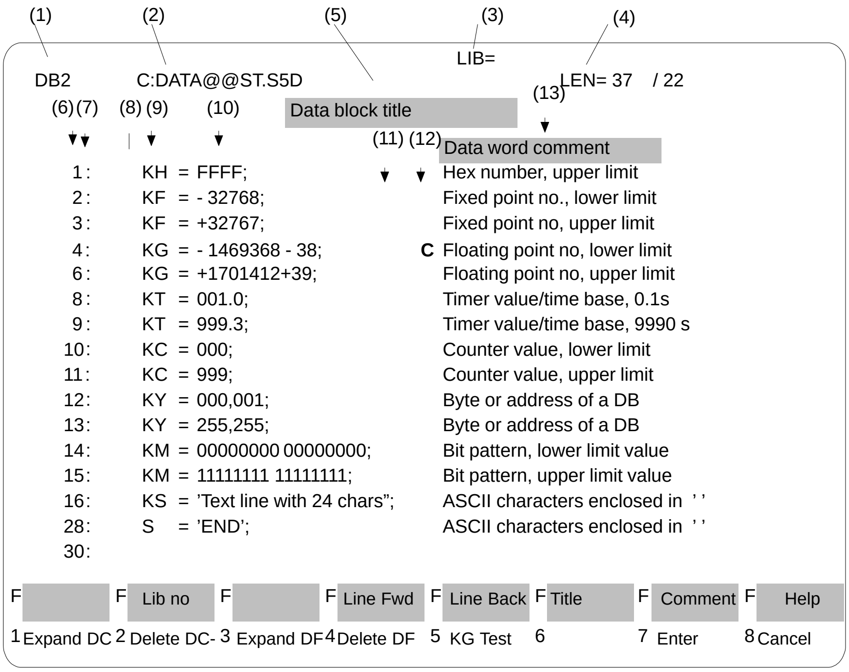


Figure 9-3 Input field of the DB Editor

Fields and Entries

The fields on the screen have the following significance:

Table 9-1 Displayed Fields

No.	Input Field	Explanation
(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 DATAxxST.S5D).
(3)	`LIB field`	In this field you can input a maximum 5-digit long DB library number (number from 0 to 99999) for the DB.
(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.
(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.

Table 9-1 Displayed Fields, continued

No.	Input Field	Explanation
(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`	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.
(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.

Function Keys The function keys of the basic menu

F1	F2	F3	F4	F5	F6	F7	F8
	Lib.No.		Line Fwd	Line Back	Title	Comment	Help
Expand DC	Delete DC	Expand DF	Delete DF	KG Test		Enter	Cancel

Key	Explanation
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.

9.2.1 Editing Block Comments

Overview

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.

Ready to Start ?

You have selected [X] *with comments* 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

Follow the steps outlined below:

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.

F	F		F	F
1 Insert	2 Delete		5 Insert L	6 Delete L
⇕				
F				
1 Overwrite				

Deleting Characters

Position the cursor on the first character to be deleted.

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

Completing/Saving a Block Comment

Press ***F8*** *= Return* or the *Insert* key.
STEP 5 displays the data block to be edited on the screen. The text input up to this point is retained. If you save the data block, STEP 5 then saves the block comment.

Press the ***Insert*** key.

9.2.2 Inputting the Block Title

Overview

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

Press ***SHIFT F6*** *= Title* or press the ***COM*** key. The cursor jumps to the input field for the block title.

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

The title is buffered, but is only saved in the comment block in the program file when you save the whole block.

9.2.3 Influencing the Length of the Block Preheader

Overview

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...).

9.2.4 Entering the Library Number

Overview

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

Follow the steps outlined below:

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.***

9.2.5 Changing Data Formats

Overview

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;

9.2.6 Inputting Data Words

Overview

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 + 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

How to Input Data Words

Follow the steps outlined below:

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. With some negative floating point numbers, rounding errors can occur.

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;

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
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’;

9.2.7 Inputting Data Word Comments

Overview

Data word comments are texts that you can enter in each line of a data format.
A data word comment is a maximum of 32 characters long and always assigned to the first data word (with format KS, S and KG). 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 [X] with *comments: yes* in the *project settings* (*Section 4.1.1*). 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

Follow the steps outlined below:

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.

3. Press the ***Return*** key.

9.2.8 Storing a Comment

Overview

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.

9.2.9 Reproducing the DWs

Overview

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 the character < 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.

9.2.10 Testing Floating Point Numbers

Overview

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.

```
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:

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***.

9.2.11 Inserting / Deleting a Line

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.

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

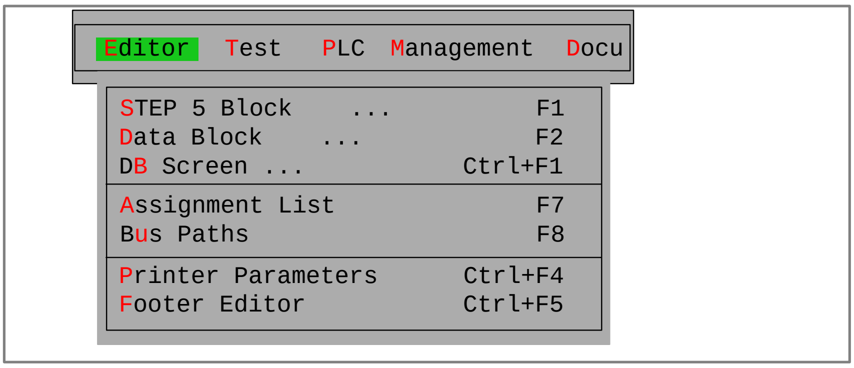
If you use ***F3*** *= Expand DF* or ***F4*** *= Delete DF*, the content of the data block can be changed when using the format KG owing to rounding up/down errors.

Editing DB Screens

10

Overview

DB screens are special data forms for the S5-135U and S5-155U. The parameters you enter depend on the CPU in the PLC. These DB screens belong to the particular PLC and do not contain comments.



Single DB Screens

The following DB screens can be used:

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 S5-155H	Defaults of some system program functions for the S5-155U, e.g. cold restart, warm restart, process interrupts etc.

Chapter Overview

10.1 Editing DB Screens

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

Session settings:

Program file	Name of the 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 **File** ▸ **Project** ▸ **Set F4**.

Starting the Editor

Editor
`DB Screen...`

Select the menu command **Editor** ▸ **DB Screen....** The *Edit DB screen* dialog box is displayed on the screen.

1. Decide whether you want to edit the block in the program file or in the PLC.
2. Type in the block, e.g. DB 1.
3. Select *DB1, I/O assignment* in the list box with ***F3*** and enter the settings with < Edit >.

The PG displays the I/O assignment dialog.

Figure 10-1 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.

Entering the Data

The cursor is in the first input field of the DB screen. Follow the steps outlined below:

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.

Insert a Line/Element

Position the cursor in the line before which you want to insert a line or element and press ***F1*** = *Insert L* or ***F2*** *Insert E*.

Delete a Line/Element

Position the cursor in the line you want to delete and press the ***delete segment*** key.

Delete a Character

Press the ***DEL*** key or overwrite with blanks.

Enter the Screen

Press the ***Insert*** key.

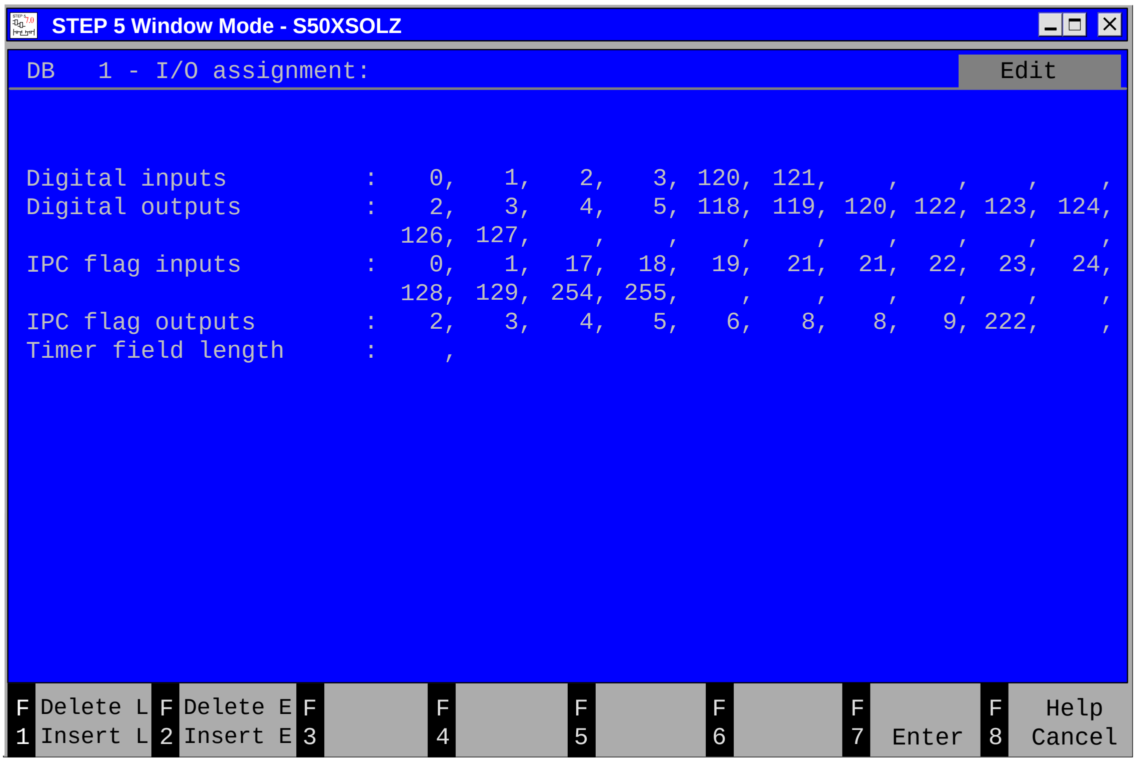


Figure 10-2 Example of a Completed DB Screen for the S5-135 U

10.2 Editing the DX 0 Screen (for the S5-135U)

DX 0 for the S5-135U

DX 0 contains the system data for the S5-135U in the form of a DB screen for this PLC. How to complete the screen is described in the programming instructions for the PLC.

Settings for the Editing Session

Session settings:

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*.

Editor
`DB Screen...`

Select the menu command **Editor ▸ DB Screen...**. The *Edit DB screen* dialog box is displayed.

Editing

Follow the steps outlined below:

1. Specify whether you want to edit the block in the program file or in the PLC.
2. Type in the block e.g. DX0
3. Select *DX0 for S5-135U.*
4. Enter your selections with the ***Return*** key. The PG displays the DX 0 screen shown below:

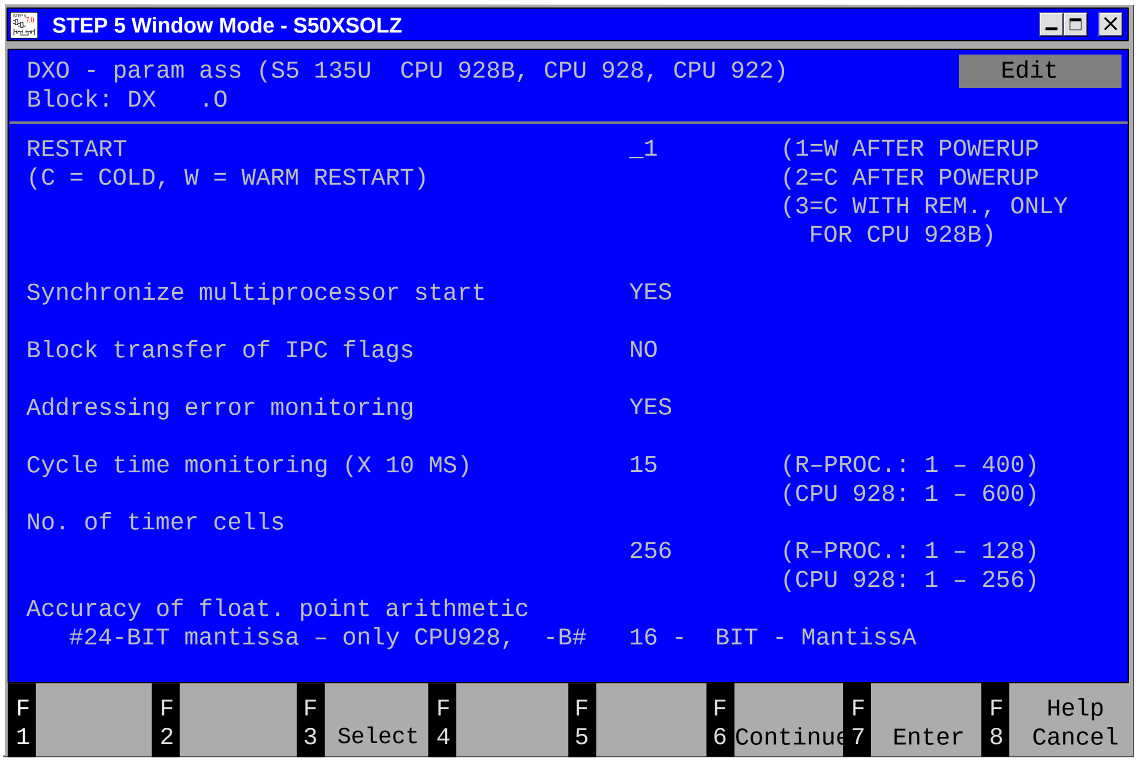


Figure 10-3 DX 0 Screen for the S5-135 U, Page 1

***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 inversed on the screen. The cursor is located in the first input field of the DX 0 screen.

DX 0 for S5-135U Page 2

With ***F6** = Continue*, you display the page 2 of the DX0 screen.

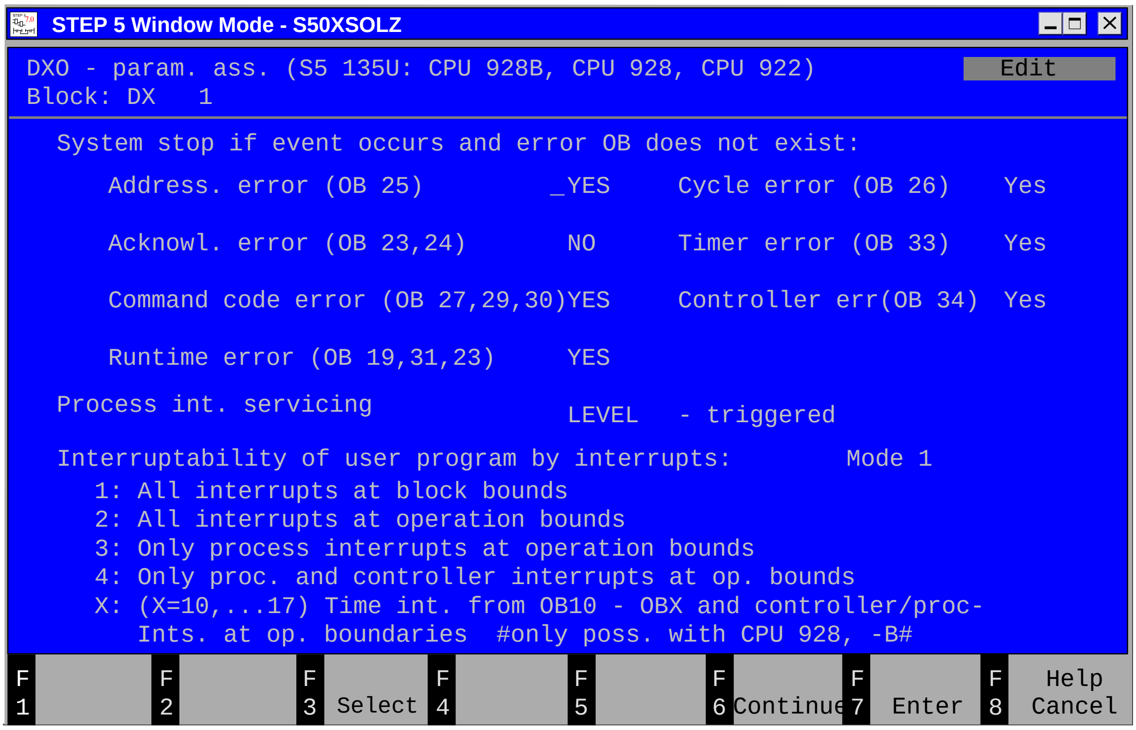


Figure 10-4 DX0 Screen for the S5-135 U, Page 2

Entering the Data

Follow the steps below:

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, 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***.

10.3 Editing the DX0 Screen (for S5-155U)

DX 0 for the S5-155U

DX 0 contains the system data for the S5-155U in the form of a DB screen for this PLC. How to complete the screen is described in the programming instructions for the PLC.

Settings for the Editing Session

Session settings:

Program file:	Name of your current program file.
Mode:	*Online*, if a PLC is connected and you want to edit in the PLC.

Editor
`DB Screen...`

Select the menu command **Editor ▸ DB Screen....** The *Edit DB screen* dialog box is displayed.

Editing

Follow the steps outlined below:

1. Specify wether you want to edit the block in the program file or in the PLC.
2. Type in the block e.g. DX 0
3. Select *DX 0 for S5-155U CPU 946/947* in the list box (***F3***) and enter the settings with < Edit >.

The PG displays the DX 0 screen shown below:

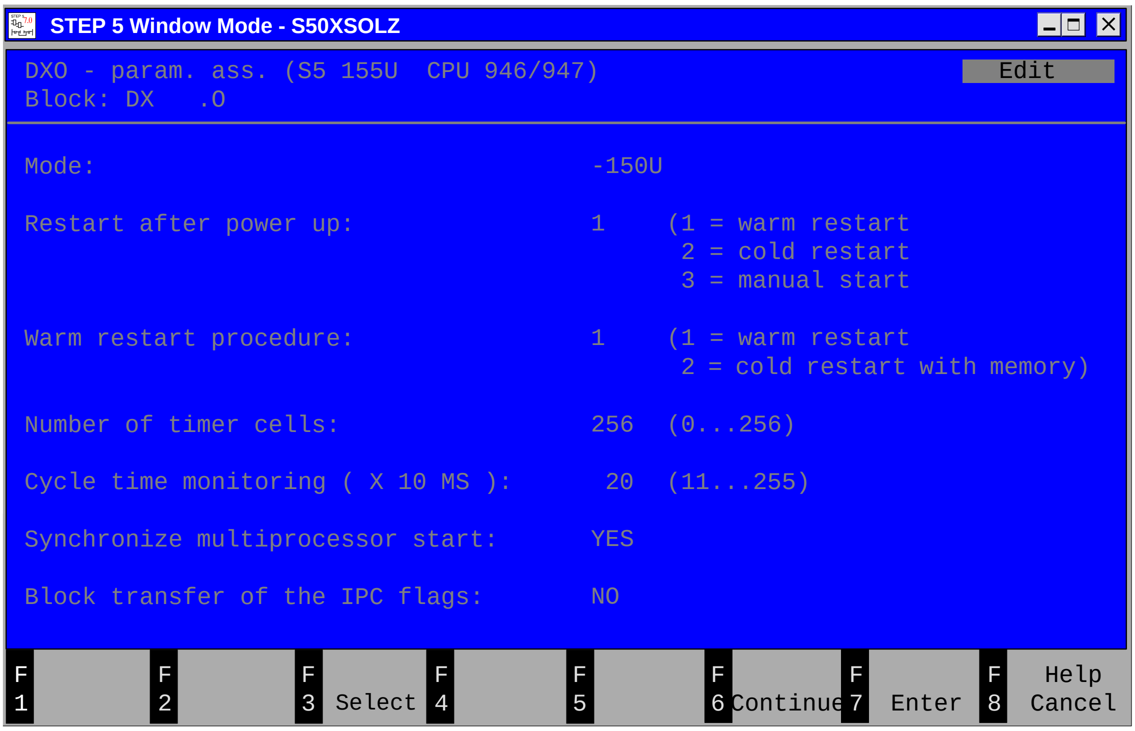


Figure 10-5 DX0 Screen for the S5-155 U, Page 1

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 or inversed on the screen. The permitted values depend on the configuration of the programmable logic controller.

DX 0 for S5-155U, page 2

With ***F6*** *= Continue*, you can display page 2 of the DX0 screen

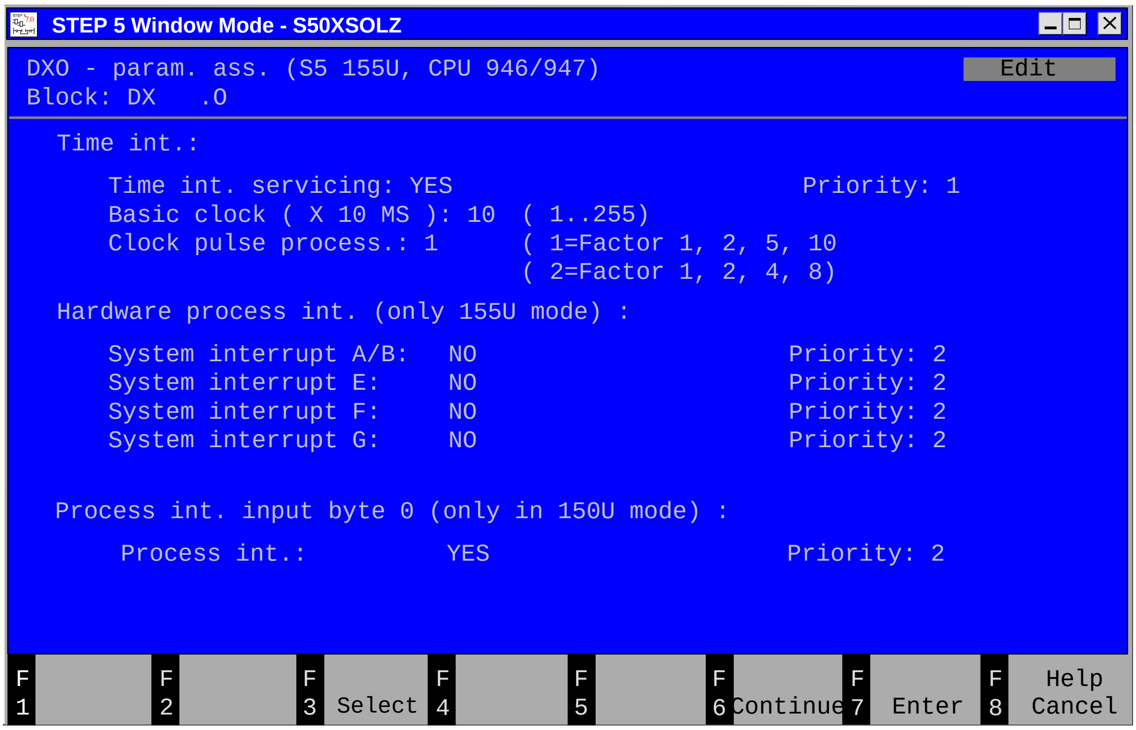


Figure 10-6 DX0 Screen for the S5-155 U, Page 2

Entering the Data

Follow the steps below:

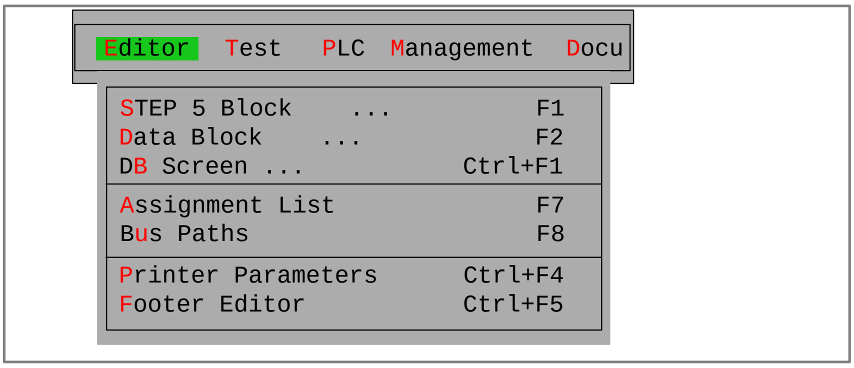
1. Position the cursor in the field in which you want to change a value, e.g. Mode ***S5-155U*** or ***S5-150U***.
2. Select the parameter with ***F3*** *= Select* or if ***F3*** *= Input* is displayed, type in the parameter using the keyboard.
3. To call page 2 of the DB screen, press ***F6*** *= Continue* and type in the parameters as on page 1.
4. To enter DX 0, press the ***Insert*** key or, to cancel your input, press ***ESC***.

Editing the Assignment List

11

Overview

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 symbols editor. While making these assignments, you can also write an operand comment for each operand.



Chapter Overview

11.1 General Aspects of Working with the Editor

Ready to Start?

You can select the length of the symbolic operands and the operand comments (**File ▸ Project ▸ Set F4**, see Section 4.1.1)

- 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 symbol or comment. The files ?????Z?.INI must first be deleted (see Section 11.4 **Management ▸ Assignment Lists ▸ Delete INI**).

The assignments and modifications to the assignments are made in the assignment list. 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 assignment list.

Calling the Editor

Editor

```
Assignment List
F7
```

Select the menu command **Editor ▸ Assignment List**. The editor for the assignment list (*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.
- → *form feed (character string PA)*

Creating an Assignment List

To create the assignment list, follow the steps outlined below.

1. You edit the assignment list as the source file (extension *Z0.SEQ).
2. The assignment list 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 occur during the conversion, STEP 5 writes the errors in an error file (extension *ZF.SEQ). You can display or print out this file 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.SEQ is also created.

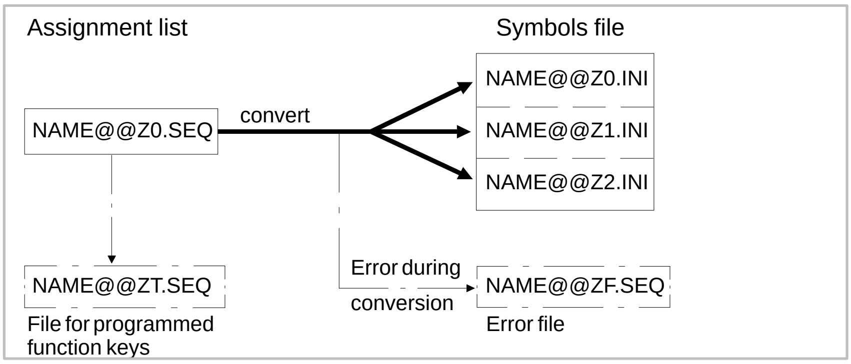


Figure 11-1 Creating an Assignment List

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 11-1 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

Table 11-2 Overview of Permitted Operand Types

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:

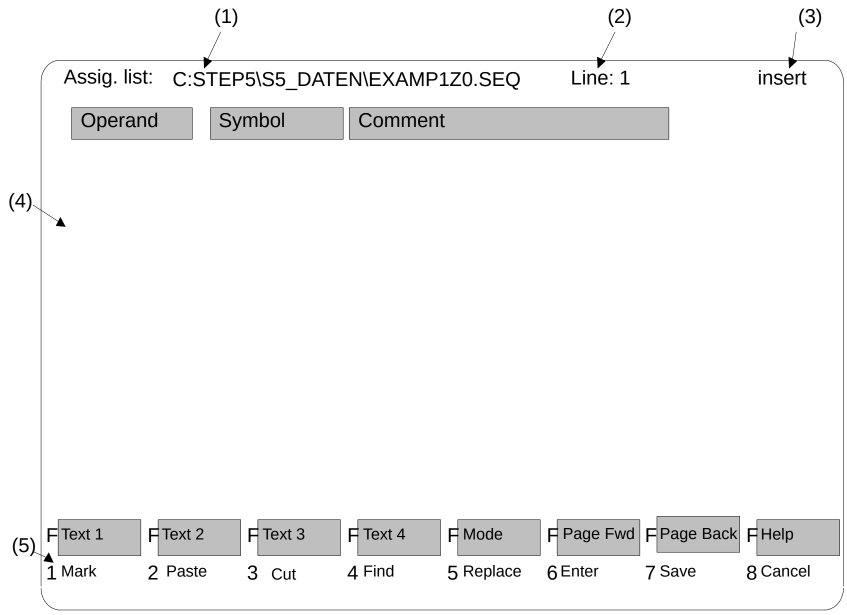


Figure 11-2 Screen Layout with Lines and Areas of the Editing Field

Screen Areas

Table 11-3 Screen Areas

Line	Explanation
(1)	Drive and name of the assignment list. The name is preset with the name of the symbols file selected in the settings. Up to 32 characters of the complete file name are displayed. If the name is longer, a shortened version is displayed.
(2)	Number of the line in which the cursor is located.
(3)	Mode display, can be switched over between insert and overwrite mode with ***SHIFT F5*** = *Mode*.
(4)	**Editing Area** 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.
(5)	**Function key** menu for calling editing functions

Function Keys

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**** *= Paste*. 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* *= Paste*	Fetches a text copied with ***F1****= Mark* and pastes it at the cursor position.
F3* *= Cut*	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 inserted using ***F2**** *= Paste*. 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 *= 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.
F7 *= Enter*	Complete the editing session and store the assignment list. The conversion to the symbols files is started automatically.
F8 *= Cancel*	Cancel the editing session without storing the assignment list.
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.

Keys with * call further key levels

11.2 Creating the Assignment List

Procedure

Type in the character string for the absolute operand, e.g. **I 1.1**.

1. Position the cursor in the symbols column using the mouse or ***TAB***.
2. 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 the project settings (see Section 4.1.1). 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 assignment list, this field remains empty. The symbolic operand is only entered in the assignment list following a conversion (→ *Management*, → Convert 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 (see Section 4.1.1). 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.

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. 11-3) can no longer be deleted by the editor. If you want to eliminate this character, you must delete the whole line (→ ***F3*** *= Cut*, ***F1*** *= Line*).

Note

Additional comments and printer control characters only exist in the assignment list. If you generate an assignment list from the symbols file using → *Management, Convert INI > SEQ*, additional comments and printer control characters (*.PA*) are lost.

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 assignment list, this control character produces a form feed in the printout. The control character is not entered in the symbols file (*Z0.INI).

Complete Editing

Follow the steps below:

1. *Press* **F7** *= Enter.*
   The assignment list 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.

Special Characters

For symbols, blanks and most special characters can be used with the exception of backslashes “\”. Illegal characters are rejected with an error message.

Note

Whenever possible do not start or end with a blank. They can hardly be recognized on the screen or in printouts!

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 assignment list, you can start a new translation by entering again.

If several errors occur, STEP 5 displays the message:
`n lines processed, m errors found. Display error list?:`

Yes: the error list is displayed
No: you exit the editor

STEP 5 records the error in the *ZF.SEQ file.

You can output this error list with the management function
→ *Assignment lists, Output Error List*.

Assig. list: C:EXAMP1Z0.SEQ   Line: 12   - Insert -

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
; Form feed with the characters		
; .PA		

Line: 1

F1	F2	F3	F4	F5	F6	F7	F8
Text 1	Text 2	Text 3	Text 4	Mode	Page Fwd	Page Back	Help
Mark	Paste	Cut	Find	Replace	Save	Enter	Cancel

Figure 11-3 Example of the Assignment List

11.3 Editing Support

Overview

STEP 5 provides editing functions when you create the assignment list and they can be activated using the function key menu. The individual functions are described below.

F1 *= Mark*

F1	F2	F3	F4	F5	
Text 1	Text 2	Text 3	Text 4	Mode	
Mark	Paste	Cut	Find	Replace	

⬇

F1	F2	F3	F4	F5	F6
					Page Fwd
Line	Text	Field Sta	Field End	File	Fct Keys

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 assignment list.

F1 *= Line* — Mark the line containing the cursor so that it can be copied elsewhere.

F2 *= Text* — Mark 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 assignment list, → ***F5*** *= File*

F5 *= File* — Save the marked field in a different assignment list. 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 *= Paste*

F	Text 1	F	Text 2	F	Text 3	F	Text 4	F	Mode
1	Mark	2	**Paste**	3	Cut	4	Find	5	Replace

F		F		F		F		F	
1	Line	2	Text	3	Field	4		5	File

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. pasted 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 assignment list 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 assignment list 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 it by generating the assignment list from the symbols file using → *Management, Convert INI > SEQ*. The conversion, however, ignores comments and control characters.

***F3** = Cut*

F1	F2	F3	F4	F5
Text 1	Text 2	Text 3	Text 4	Mode
Mark	Paste	**Cut**	Find	Replace

F1	F2	F3	F4	F5
Line		Field Sta	Field End	

With this function you can delete a line or field. The deleted line or field is written to the buffer. If you have already copied a field or line to the buffer this is overwritten. You can then copy the content of the buffer elsewhere → ***F2** = Paste*.

***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*

F1	F2	F3	F4	F5
Text 1	Text 2	Text 3	Text 4	Mode
Mark	Paste	Cut	**Find**	Replace

F1	F2	F3	F4	F5	F6	F7
					Page Fwd	Page Back
Text +	Text –	Operand+	Operand –	Line	To Start	End

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*

F1	F2	F3	F4	F5
Text 1 / Mark	Text 2 / Paste	Text 3 / Cut	Text 4 / Find	Mode / **Replace**

F1	F2	F3	F4	F5
Conf.	No Conf		All	

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 *project settings* (see Section 4.1.1).

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

F       F       F       F       F       F       F
1       2       3       4       5       6       7 Enter
```

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***.

11.4 Modifying the Assignment List

Overview

If you want to modify an assignment list you have already created and translated, you can edit the assignment list providing it still exists. If the assignment list does not exist, this is generated automatically from the symbols file and output.

Remember 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, there are two ways of doing this:

1. You can create a new assignment list (**File** ▸ **Project** ▸ **Set F4**, tab 3) and copy the existing assignment list to this new file using the editing functions ***F2*** = *Paste* and ***F5*** = *File*.
2. You can delete the files ??????Z?.INI (**Management** ▸ **Assignment Lists** ▸ **Delete INI**). You can then increase the symbol or comment lengths in **File** ▸ **Project** ▸ **Set F4** (tab 3). When you next start the editor, it uses the new settings.

How to Modify and Change the Field Lengths

Follow the steps below:

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*** *= Paste* 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 (Insert) with ***F8*** = *Return*. You can now edit the assignment list 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 assignment list with the same name are overwritten without prompting you to confirm your intention.

Bus Paths

12

Overview

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.

```
 Editor   Test   PLC  Management   Docu

   STEP 5 Block    ...          F1
   Data Block    ...            F2
   DB Screen ...            Ctrl+F1
   -------------------------------
   Assignment List              F7
   Bus Paths                    F8
   -------------------------------
   Printer Parameters       Ctrl+F4
   Footer Editor            Ctrl+F5
```

Chapter Overview

12.1 Bus Paths

General

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** (**File ▸ Project ▸ Set F4**) 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** (**File ▸ Project ▸ Set F4**) 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.

Assignment Path > File

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*).

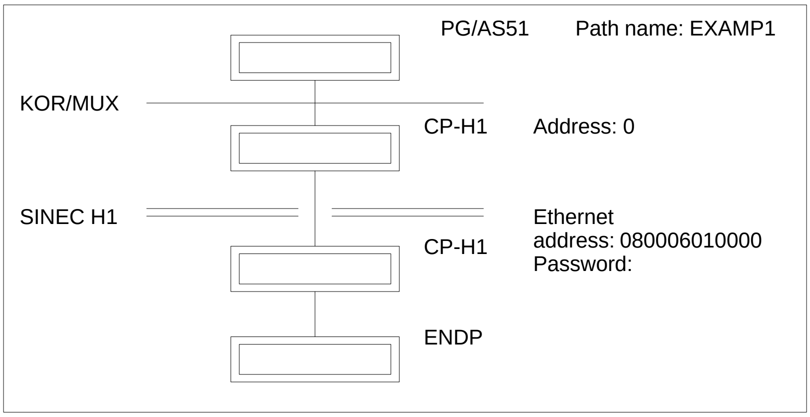


Figure 12-1 Example of an Edited Path

12.2 Editing a Bus Path

Settings

The interface for the start node (AS511, H1 or L2) must be set. For more information about the *project settings* refer to Section 4.1.1.

Starting the Editor

Editor
`Bus Paths F8`

After selecting the *Bus path* function the *Select Bus Path* dialog box is displayed. You can set the following:

- Path file
- Path name

Setting Bus Paths

The inputs you can make in the *Select Bus Path* dialog box are described in the following table:

Key level 1	2	Explanation
F1		Edit: The path editor is started. You can now edit the bus path in the working field displayed. The function keys are assigned a new function.
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	Cancel: Returns to the last menu (select function).
F3		Select: The *Path file and path name* dialog box is displayed. This lists all the path files and path names. You can enter the path or file marked by the cursor.
F4		Activate: With this function, the set path is displayed. You can correct through to the end point step by step using the function ***F3*** *= Single* or in one step (***F5*** *= All*). 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		Cancel: 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 ***F6***, beginning with the last node. Use the function 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, CP L1 and CP 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 CP 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.
F4	PG/CP-L2

During editing, these start nodes are not dependent on the set interface. The functions of the function keys 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 = *Del Elem*	Deletes the last node and/or bus from the path.
F7 = *Enter*	The edited path is saved. Step 5 returns to the previous level. With ***F3*** and the cursor on the *path file or path name* input field, the *Select path file and path name* dialog is displayed with all the path names and path files
SHIFT F7 = *Files*	You can edit the files assigned to this path.
F8 = Cancel	Return to the last menu without saving.
SHIFT F8 = *Help*	Displays information about the functions of the function keys at the current level.

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
F1		**Proj Sett** The file names from the project settings are used.
F3		**Select** The *select file* 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 ***Enter***
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 canceled.
F4		**80/132 C** 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). With ***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.

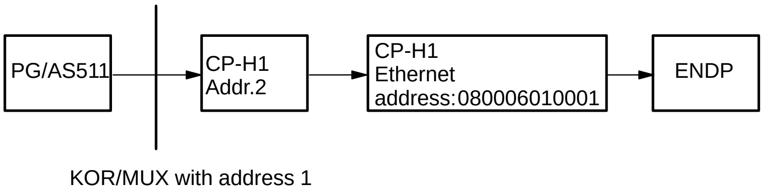
If the error message `Drive and project settings different` appears or if `Specify drive from project settings` is displayed as the directory, the drive identifiers of the files must match those in the project settings, before the selected files can be entered in the currently set project (path option in the project settings set to *Confirm* or *Always*).

With ***F1*** *Project Setting,* the files can be entered in in the path file from the currently set project and the file names can then be edited or selected (F3).

12.3 Example

Task

You want to create the following path:



The AS511 interface is set. You have selected the function ***Editor, Bus Paths F8***.

Operation

The *Select Bus Path* 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 function keys:

   F2 = PG/AS511

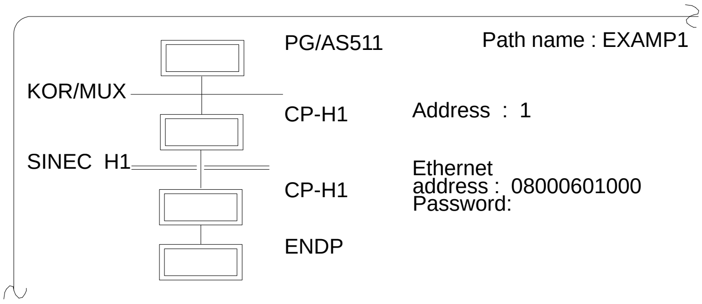
   F3 = PG/CP-H1

   F4 = PG/CP-L2
4. Press ***F2*** *= PG/AS511.*
5. Press ***F2*** *= KOR/MUX*

   The KOR/MUX bus is added .
6. Press ***F3*** *= CP-H1*

   The CP-H1 node with the SINEC H1 bus is added.
7. Press ***F3*** *= CP-H1*

   The CP-H1 node with the SINEC H1 bus is added.
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. The screen should appear as follows when the path is complete.



The bus path has been edited completely. The path must now be stored.

10. *Press **F7** = Enter.*
    The path is stored in the path file and you can activate it at any time.

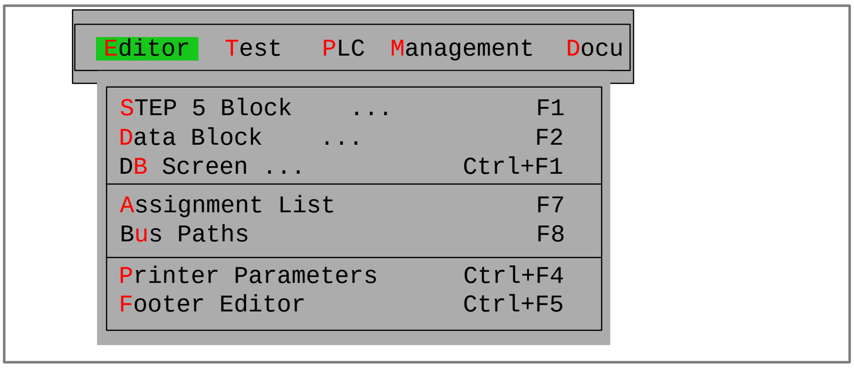
Printer Parameters

13

Overview

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
- Select and edit the footer and enter text (see Chapter 14).



Chapter Overview

13.1 Setting Printer Parameters

Overview

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 many printer types. These contain settings for specific printers and the type of printout (portrait, landscape). In the *project settings*, you can double-click *Printer file* to obtain a list of printer files (*DR.INI) available in the system directory or press ***F3*** to display a printer list box.

Setting

Select the printer file of type *DR.INI in the *Documentation* tab page of the project settings. The asterisk (*) stands for the six-character name of the printer file. For more information about settings, refer to Section 4.1.1.

Setting Parameters

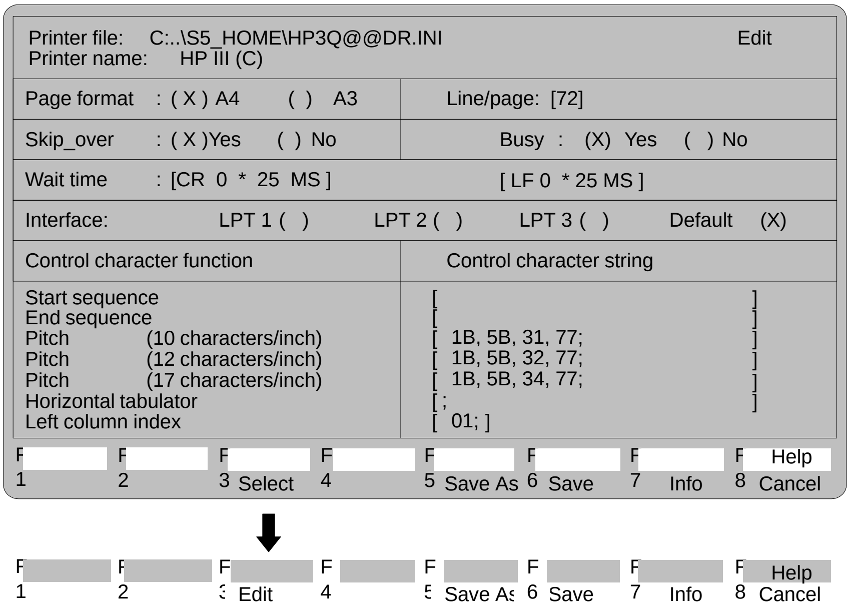
Editor

`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 the *Printer parameters* dialog.

Dialog Box

The p*rinter parameters* dialog is displayed (example below). The file C:HP3Q@@DR.INI for the HP III (C) printer was selected in the *Documentation* tab page of the project settings.



Parameters 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 port 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.
Control character 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
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.

Printer names Explanation of the printer names in the shipped printer files.

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.

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 character 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, key assignment*).
F8 *= Cancel*	Return to the calling level.

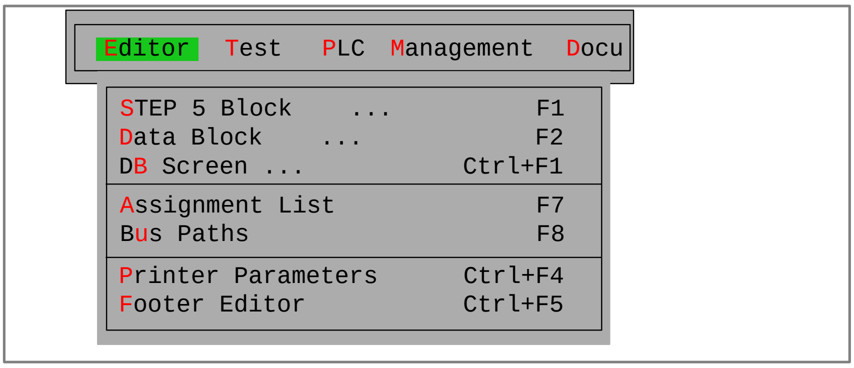
Footer Editor

14

Overview

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, for example:

- SIMATIC S5
- Program file
- Block
- Segment
- Page number



Chapter Overview

14.1 Editing Footers

Settings

Select a footer file of type *Fx.INI with the menu command **File ▸ Project ▸ Set** (Documentation). The asterisk stands for a six character name and x stands for 1 (80 characters wide) or 2 (132 characters wide). For more information about the settings, refer to *Section 4.1.1*

Starting the Editor

Editor
`Footer editor`

When you start the footer function, an editing window is displayed with a footer determined by the size you have selected. 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 automatically. 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*:
When you print using the enhanced mode (KOMDOK), this is overwritten by the current system date.
Fields with ### entered cannot be overwritten.

Editing Window

The screen below illustrates the editing window for 132 character wide footers. 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 (`NEW`) or whether you are modifying (`EDIT`) an old one.

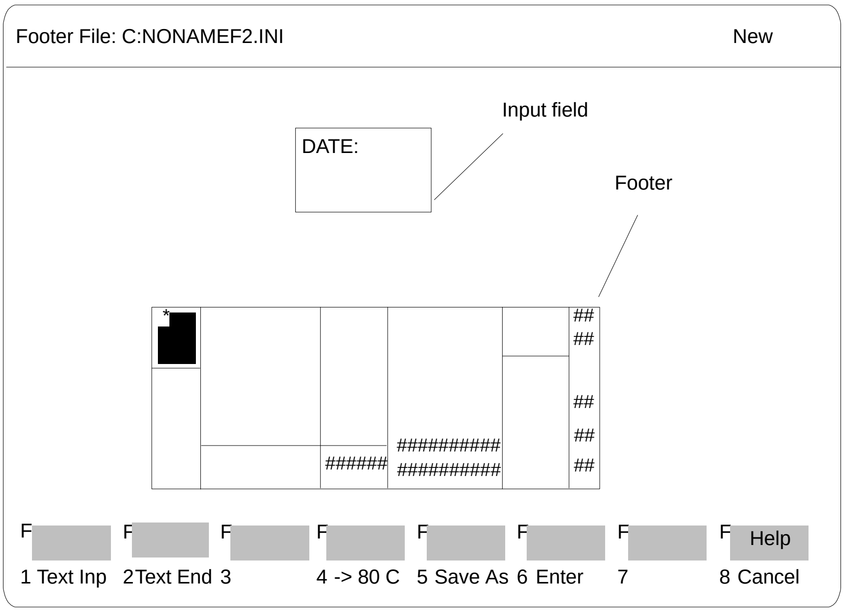


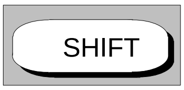
Figure 14-1 Editing Window for 132 Character Wide Footer

Function Keys

With the keys ***F1*** to ***F8***, you can activate the following functions:

Key	Function
F1 = *Text Inp*	Input text in the window displayed above
F2 = *Text End*	Complete text input
F4	80 C. Switch to a footer width of 80 characters 132 C. Switch to a footer width of 132 characters
F5 = *Save As*	When you press this key, the *Save footer file as* dialog is displayed. The cursor is located in the *Footer file* field. You can select a file with ***F3*** or by double-clicking.
F7 = *Enter*	You save the modified footer file
F8 = *Cancel*	Cancel and return to the previous level.
SHIFT F8 = *Help*	

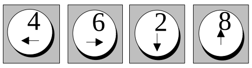
Cursor in the footer: (***SHIFT*** + *a* ***cursor*** *key*)



(4) Positions the cursor in the next footer field to the left.

+

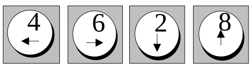
(6) Positions the cursor in the next footer field to the right.



(2) Positions the cursor in the footer field below.

(8) Positions the cursor in the footer field above (also without SHIFT).

Cursor in the input window



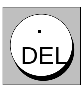
(4) Positions the cursor on the previous character .

(6) Positions the cursor on the next character.

(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.

Delete character



The character marked by the cursor is deleted and the remaining characters are shifted together to close the gap.

Part 3:
Test, Management, Documentation

Test

15

Overview

This submenu includes test, information and start-up functions that you can execute on the PG in the online mode.

```
Test   PLC   Management   Documenta

Block Status      ...   Shift + F6
Status Variable         Shift + F7

Force Variables         Shift + F8
Force Outputs           Shift + F5

Program Test            ON
Program Test            OFF
```

Requirements

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.

Warning

Bus connections and plug connections may not be disrupted while online functions are active.

This may result in serious functional errors, such as causing the PLC to STOP, or the programming device to crash.

Chapter Overview

15.1 Online Functions

Overview

The following table provides you with an overview of the possible online functions. 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 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 variables[1 2]	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)
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).
[2] Force variables is also possible offline to allow you to edit variables blocks.

15.2 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.

Input

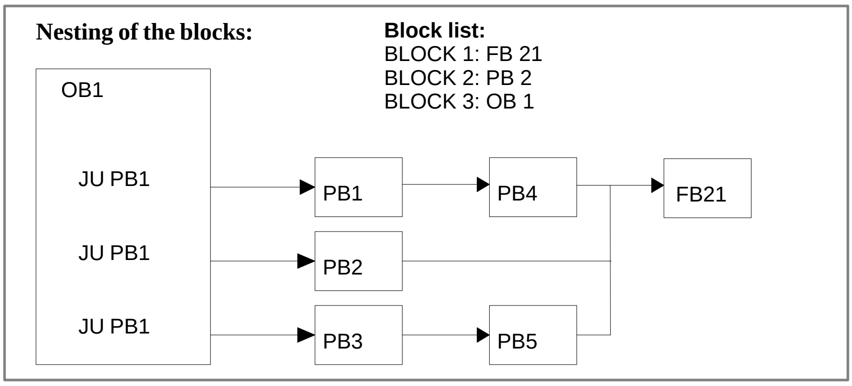
After you select the function in the Test menu, the *Block status* list box appears. Here, you specify the block to be tested (→ *User interface, List box, Section 3.6*).

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.
`Assignment list`	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



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, it is not one of the 20 displayable statements; the number of statements depends on the PLC).

Example for CSF

After ***OK***, STEP 5 begins the status processing and displays, for example, the following screen in CSF:

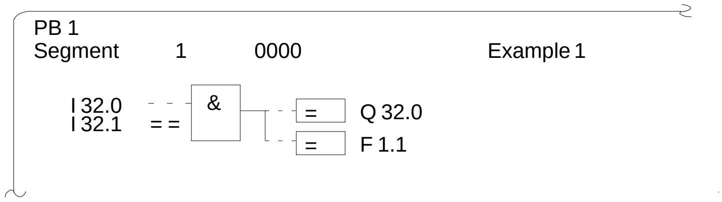


Figure 15-1 Status Processing

The display is **not** updated following each cycle.
All the functions made available in the function key menu (→ *Editor, STEP 5 blocks, Section 5.1*) can be executed during status processing.

Note
You cannot display addresses.

Example of STL

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
    :***
```

```
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. 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.

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

Abbreviations

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

Block Status Processing

This consists of the following actions:

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?`*

Possible messages:	Causes:
`Statement not processed`	– block is not called – statement is skipped – a block or sequence of blocks does not exist – PLC in the STOP mode
`Block does not exist in PLC`	– he block to be tested does not exist – the block to be tested calls a further block that does not exist in the PLC.

15.3 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, 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.

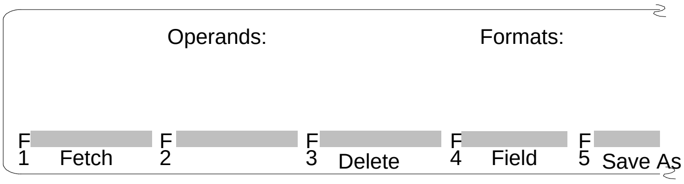


Figure 15-2 Table for Editing the Operand List

Available in the Submenu

Key	Function
***F1** = Fetch*	Call a variables block
***F3** = Delete*	Delete the current line
***F4** = Field*	Display variables in fields, keys + or – fetch the previous or next field.
***F5** Save As*	Save the operand list as a variables block
***F6** = Activate*	Activate status processing (= enter). Only available when at least one operand is entered
***F7** = Save*	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

Changes to the Operand List

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 (***F5** = Save As* 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*.

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.
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(Sich_als)*, you do not specify a variables block number. The function is only available when a variables block is selected.
Store operand list	1. *Press **F5** = Save As* 2. Fill in the command line Save 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: e.g. 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.

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 15-3 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/abort status 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

15.4 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. Select **Test ▸ 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. 15-3) or the variables block last selected for *force variables*.

Editing the Operand List

Operand	Permitted data formats	
F/Q/I/S[1)]	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/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 displayed (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.

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 (see Section 15.3).

15.5 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. 15-3) or the variables block last selected for *force variables*.

Editing 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 editing options are explained in Section 15.3.

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***,** STEP 5 returns to the basic menu of the functions. Refer to Section 15.3 *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.

15.6 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 interrupt STACK
- Info about the block 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. a single 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 Section 15.2 Block Status **)**.
   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,. To carry out a correction while the program test is still active, 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. The basic menu shows whether or not the program test is activated.

15.7 Program Test OFF

Test

`Program Test OFF`

This function deactivates the program test.

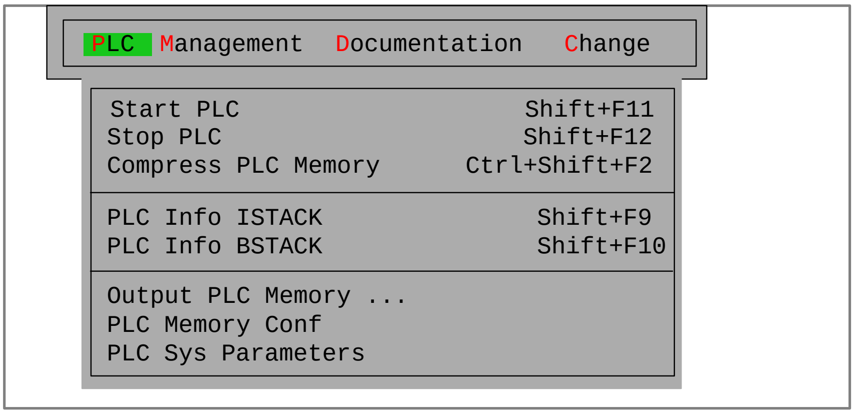
Select **Test ▸ Program Test OFF**. The PLC changes to the STOP mode and must be restarted (→ *PLC, Start PLC*) or by changing the CPU selector from STOP to RUN).

PLC

16

Overview

Within this menu, you can start and stop a PLC connected online and compress the user memory in the PLC.



Chapter Overview

16.1 Starting the PLC

PLC
`Start PLC`

The *Start PLC* function triggers a cold restart or warm restart on the programmable controller. 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.

If errors occur, this is indicated by messages. The particular message depends on the CPU you are using.

16.2 Stopping the PLC

PLC
`Stop PLC`

The *Stop PLC* function switches the programmable controller to the STOP mode. 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.

The messages displayed depend on the CPU you are using.

16.3 Compressing the PLC memory

PLC
`Compress PLC 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.

16.4 PLC Info ISTACK

PLC
`PLC Info ISTACK`

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.

Figure 16-1 Table of Control Bits (for example 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***.

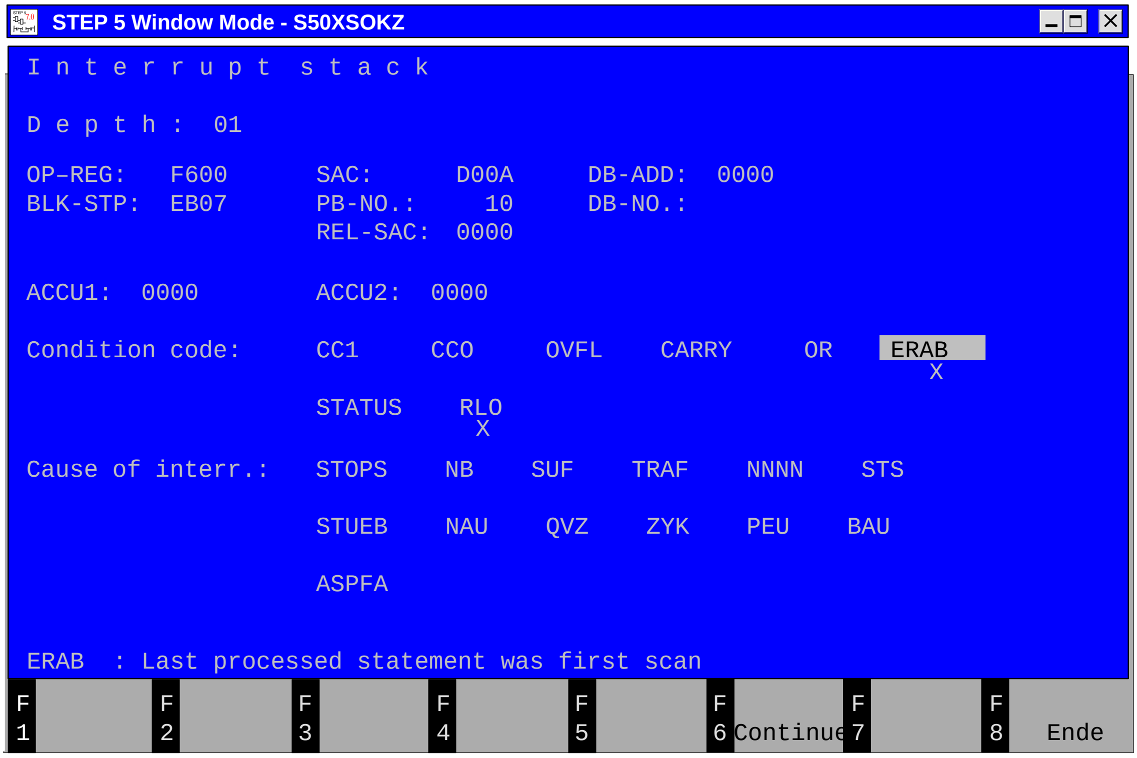


Figure 16-2 Display of the Interrupt Stack

With ***F2***, you can jump directly to the interrupt point and, if required, edit the program.

Note

There may be more than one screen page.

16.5 PLC Info BSTACK

Function

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 address at which the program must be continued once the newly called block has been processed.

PLC
`PLC Info BSTACK`

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		

Figure 16-3 Block Stack

Possible message:

1. `Wrong mode at PLC`

The PLC is not in the STOP mode.

2. `Empty or incomplete stack.`

16.6 Output PLC Memory

Function

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.

PLC
`Output PLC Memory`

Select the menu command **PLC ▸ Output PLC Memory.** The job box *Output PLC memory* is displayed. You can browse through the box and make your selections.

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. Click Output.
   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 ***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.

16.7 PLC Memory Configuration

PLC
`PLC Memory Conf`

This function outputs the absolute addresses and their contents on the screen, to a printer or to a print file.

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.

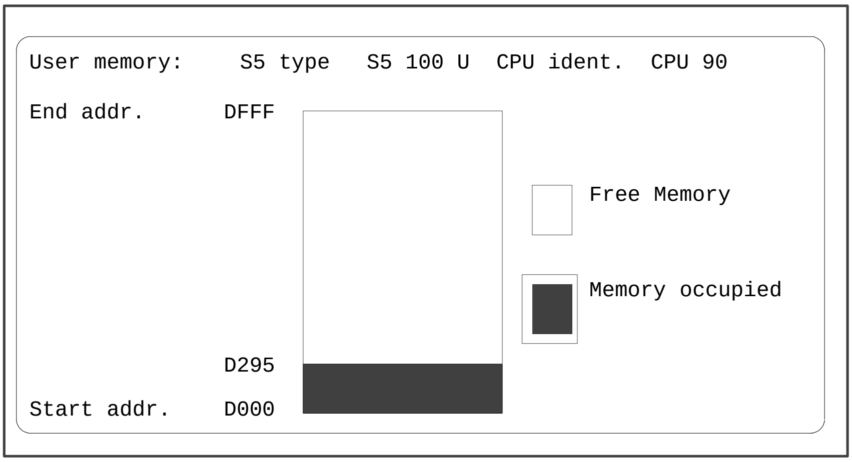


Figure 16-4 Size of the User Memory and Memory Occupied in an S5-100U

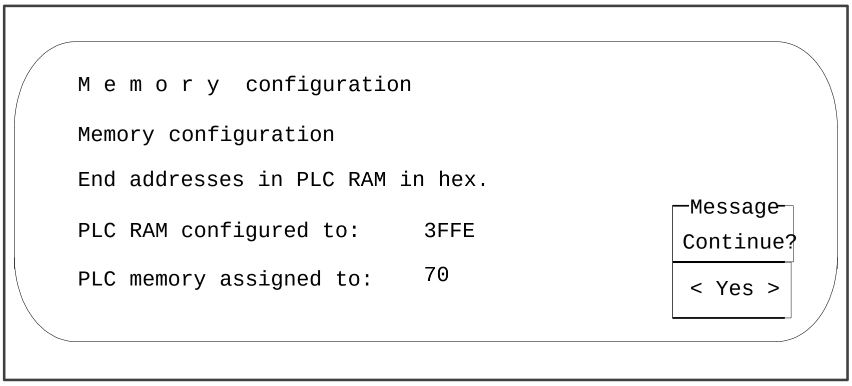


Figure 16-5 User Memory Size and Assignment as Text

16.8 PLC System Parameters

Function

With this function, you can display the following PLC system parameters on the screen:

- CPU identifier
- CPU type
- CPU number
- memory distribution
- block list lengths

PLC
PLC Sys Parameters

Select the menu command **PLC ▸ PLC Sys Parameters.** The job box *PLC system parameters* is displayed.

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.

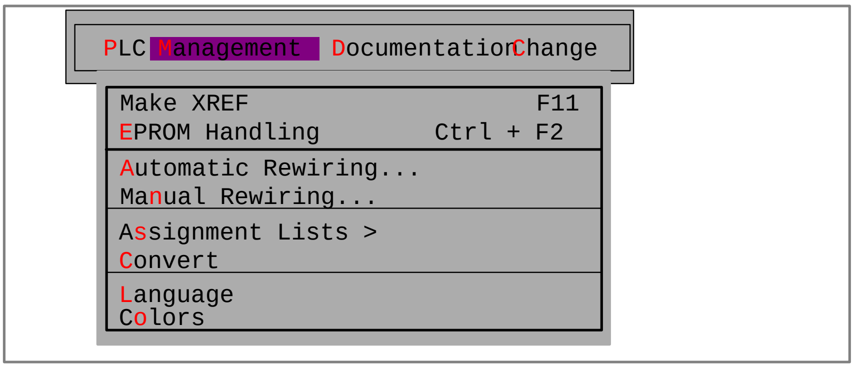
PLC software release	Z 01
CPU identifier	S5 100 U CPU 90
PGAS software release	Z 00
I/O module inputs	0
I/O module inputs	0
Process image inputs	EF00
Process image outputs	EF80
Flag memory	EE00
Timer memory	EC00
Counter memory	ED00
RS memory area	EA00

Management

17

Overview

This main menu includes a series of utilities.



Chapter Overview

17.1 Make XRF

Function

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.

Management
`Make XRF F11`

Select the menu command **Management ▸ Make XRF**

After triggering the function in the main menu, this function is executed automatically.

The 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.

17.2 EPROM Handling

Function

With this function, you transfer STEP 5 blocks from a program file to an EPROM/EEPROM. This is commonly known as *blowing* an EPROM.

These memory submodules must be inserted in an EPROM port on the PG.

STEP 5 supports you in selecting the correct 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.

Management
`EPROM Handling`

Select the menu command **Management ▸ EPROM Handling *Ctrl+F2.*** As soon as you have selected this function, the *EPROM programming* dialog appears.

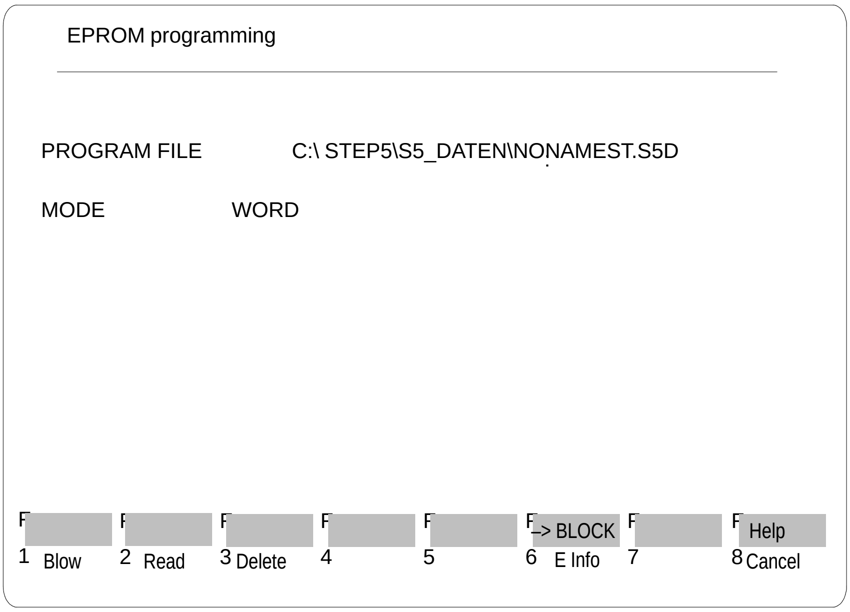


Figure 17-1 EPROM Programming

The program file you selected in the *Blocks* page of the project settings is displayed and cannot be changed here. The mode you selected in the EPROM page of the project settings can also be selected here with ***SHIFT F5***. You activate the individual functions with the F keys in the function key bar.

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 ***F6*** *= Mode.*
- Press ***F1*** *= Blow.*
- Press ***F12*** *= Help* for block information.

The command line is then displayed at the lower edge of the screen. 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 17-2 Example of the EPROM/EEPROM Information Screen

How to Activate Functions

The various EPROM functions activated by function keys (***F1*** to ***F8***) are explained in the following table.

Key level 1	Key level 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 functionEnd 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 functionEnd address  nnnnnnnn` EPROMs are erased with an erasing unit.
F5		**E Info:** Displays information about the submodule inserted in the EPROM slot. Changes to the next key level.

Key level 1	Key level 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.
	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* 17-2.
	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	**Help** Display function key assignment
F8		**Return** Return to function selection

17.3 Automatic Rewiring

Overview

With the *rewiring* function, you can rename operands as follows:

- automatically, based on an two symbols files or
- manually, based on a list of changes you have created (see *Section 17.4)*

You copy the symbols file belonging to the user program and change the addresses of the required operands in this file.

The PG uses this new symbols file as a reference list to find the changed operands automatically in the entire old user program (or in individual blocks) and to save the renamed operands in the second program file as a “new user program”.

The “old” program file is retained if the source and destination files are different. You can modify any number of operands.

Rules

You can select these operands belonging to the types input I, output Q, flag F, timer T or counter C in symbolic or absolute format. ***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.

Data blocks cannot be symbolically rewired. To transfer the structure of the user program unchanged, the data blocks must be transferred separately to the new file.

Example

The symbols **–Flag0** and **–Inp0** in the symbols file SYMOLDZ0.SEQ are assigned to the operands **F.0.0** and **I0.0** in the program file REWOLDST.S5D.

In a new symbols file SYMNEWZ0.SEQ, the symbols **–Flag0** and **–Inp0** are assigned to the operands **F 1.2** and **I 2.0**.

By automatically rewiring, all the same symbols (in SYMOLDZ0.INI and SYMNEWZO.INI) are assigned to the new operands in the new program file REWNEWST.S5D .

Note

If you replace **I1.0** with **I20.0, IB/IW1** does **not** become **IB/IW20**.

Management

```
Automatic
Rewiring ...
```

Select the menu command **Management ▸ Automatic Rewiring..**

After you call the function, the PG displays the *Automatic rewiring* 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 (see *Section 3.9)*.
After clicking **<Rewire>**, STEP 5 outputs a list of the files affected by the renaming function either on the screen, printer or to a file.

Errors

If an error occurs during the rewiring, 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.

17.4 Manual Rewiring

Function

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.

Management

```
Manual Rewiring
...
```

Select the menu command **Management ▸ Manual Rewiring...** After you have selected the function, the PG displays the job box on the screen.

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.

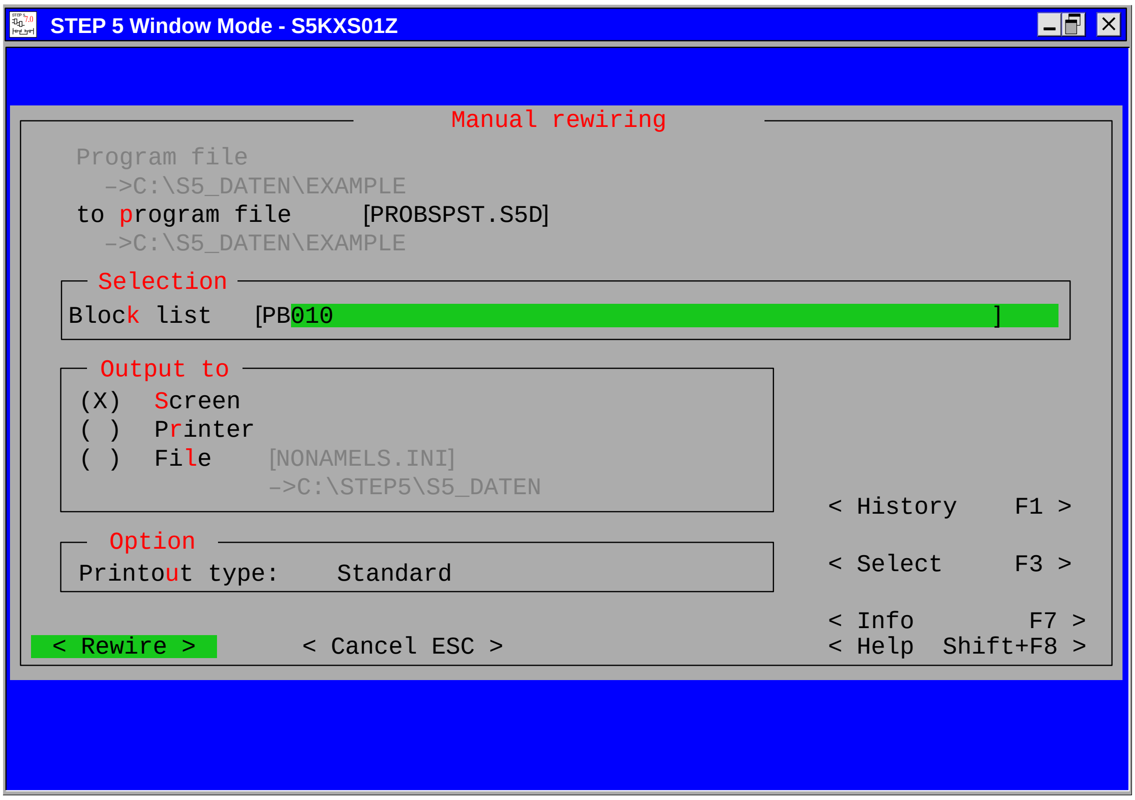


Figure 17-3   Dialog for Manual Rewiring (Example)

After you click ***<Rewire>***, STEP 5 displays the empty table *Rewire manual* 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. Complete each entry with the ***Return*** key.

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 (see *Figure 17-4*).

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 to 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  r e w i r i n g                                         page 1

Old program file:      WASCHAST.S5D     New program file:       PROBSPST.S5D
 –>C:\S5_DATEN\EXAMPLE                    –>C:\S5_DATEN\EXAMPLE
      Old operand:    Q 32.0                 New operand:   Q 1.1
      Old operand     F  10.2                New operand:   I   7.5

PB  2                                        LENGTH=     1
Number of rewirings:
PB  10                                       LENGTH=     2
Number of rewirings:
PB  11                                       LENGTH=     0
Number of rewirings:
```

Figure 17-4 Printout Following Manual Rewiring (Example)

To Stop the Function

Press ***ESC*** = *Cancel*

The PG does not store the block currently being processed.

Errors

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.

17.5 Assignment Lists

Function

With this function you edit the assignment lists required to address operands symbolically in your user programs.

The following functions are available:

- Translation of an assignment list into a symbols file (*Z0.SEQ → *Z0.INI).
- Translation of a symbols file into an assignment list 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 an assignment list (Convert stage V1.x V2.x).
- Deleting an assignment list 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 Chapter 11.
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.

17.5.1 Convert SEQ > INI

Function

With this function, you translate the assignment list into the corresponding symbols file.

Management
```
Assignment Lists
  Convert SEQ →
  INI
```

Select the menu command **Management ▸ Assignment Lists > Convert SEQ–>INI.....**

After selecting *Convert SEQ → INI*, STEP 5 displays the *Convert assignment list SEQ –> INI* job box in which you type in the name of the source file to be translated. If you have included absolute operands without corresponding symbolic operands in the assignment list, the following message is displayed:

`Accept absolute operand as symbol?`

Acknowledge this message either with *yes* or *no*.

If the conversion is error-free, the following message is displayed
`n lines processed, no error found`
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***.

Note

If you created an assignment list with English mnemonics for the absolute operands (Z0.SEQ), the operands will still be output in English if you output the file in another language. If you want the operands output in, for example, German mnemonics, you must delete the English assignment list and convert the symbols file back the assignment list (INI → SEQ).

17.5.2 Convert INI > SEQ

Function

With this function, the symbols file is converted to the corresponding assignment list, and sorted according to absolute parameters, symbols or as in the symbols file, as you require.

Management
```
Assignment Lists
  Convert INI ->
  SEQ ...
```

Select the menu command **Management ▸ Assignment Lists > Convert INI–>SEQ...** After you select this function, STEP 5 displays the *Convert symbols file INI –> SEQ* 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 **Compile**, the file is translated.

The conversion is completed with the message

`n lines processed, no errors found`

which must be acknowledged with ***OK***.

Note
When an existing assignment list (SEQ file) is sorted ”according to absolute parameters” or ”according to symbols”, all additional comments (;), empty lines and indentations (.PA) are lost.
When the list is sorted ”as in the symbols list”, only the additional comments remain intact.

Note
During sorting, all the control commands (.PA) and empty lines and comment lines (;) are lost.

17.5.3 Correct INI

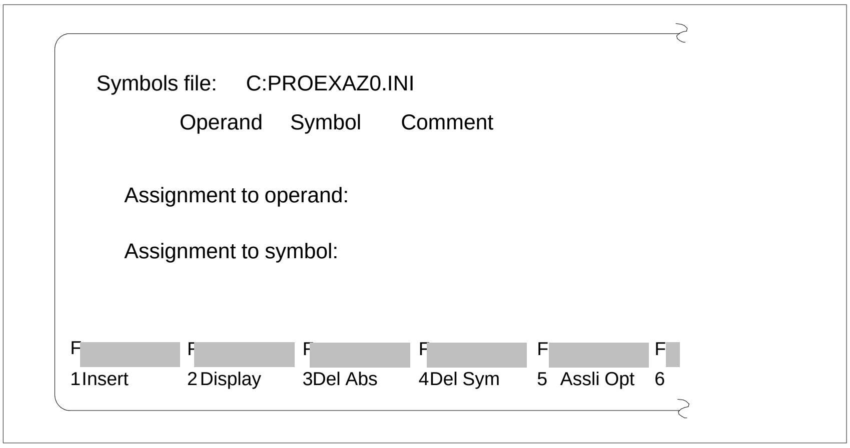
Function

With this function, you can correct individual assignments in long assignment lists (avoiding long conversion times required for all the assignments).

Management
`Assignment Lists`
`Correct INI`

Select the menu command **Management ▸ Assignment List > Correct INI.....**

After selecting this function, STEP 5 displays the *Correct symbols file* job box in which you can type in the name of the symbols file to be corrected. After clicking ***Correct***, the following box is displayed:



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*** = *Delete* 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 *= Assli Opt*	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.*

17.5.4 Convert V1.x and V2.x

Overview

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 version V3.x and higher, the byte address is four bytes long owing to the introduction of new flags (S). For this reason, the “old” symbols file must be converted to a “new” source file before you can work with it.

Assignment lists created with higher versions do not need to be converted.

Management

```
Assignment Lists
  Convert V1.x
  and V2.x
```

Select the menu command **Management ▸ Assignment Lists > Convert V1.x and V2.x**.Type in the name of the assignment list in the displayed job box. When you click ***Compile***, the file is converted.

If you have specified absolute operands without corresponding symbolic operands in the assignment list, the following message is displayed: `Acknowledge the message to suit your requirements`.

17.5.5 Delete SEQ

Management

```
Assignment Lists
  Delete SEQ
```

With this function you can delete an assignment list. At the same time, the error list file and key assignment file assigned to the file are also deleted.

After you start the function *Delete SEQ*, STEP 5 displays a job box in which you type in the name of the assignment list to be deleted if it is not already displayed.

After clicking ***Delete***, the *SEQ files are deleted. On completion of the function, the deleted files are listed on the screen.

17.5.6 Delete INI

Management

```
Assignment Lists
  Delete INI
```

With this function you can delete the symbols files (*Z0.INI, *Z1.INI, *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 ***Delete***, the symbols files are deleted. On completion of the function the deleted files are listed on the screen.

17.5.7 Output error list

Management

```
Assignment Lists
  Output Error
  List
```

STEP 5 collects the error messages occurring during one of the following conversions.

- Conversion of the assignment list *Z0.SEQ into the symbols files (*Z0.INI, *Z1.INI, *Z2.INI)
- Reconversion of the symbols files into the assignment list (INI → SEQ).

After calling the function *Output ErrorLlist* a job box is displayed in which you type in the name of the error file (*ZF.SEQ) to be output and where you want it output to (screen, printer or file). Click ***Output*** to start the function.

Example

```
File     C:\S5_DATEN\DEFAULT\PROBSPZ0.SEQ

Translation Assig. list 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 17-5 Error List after Editing the Assignment List (Example)

An error message indicates the incorrectly assigned operand, the location of the error and the error type.

Each time you translate the same assignment list, STEP 5 automatically overwrites the previously stored error list.

The file is also generated if no error occurs.

17.6 Convert

Function

This function converts project data and user files from the file format of STEP 5/ST version 6.x to the format of version 7.x. The new file format contains complete DOS paths. The following conversions are possible:

- PJ ▸ PX project file from version 6.x to version 7.x
- PX ▸ PJ project file from version 6.x to version 7.x
- PJ+AP ▸ PX project file from version 6.x (taking into account the files to which bus paths are assigned) to version 7.x

Management
`Convert ...`

Select the menu command **Management ▸ Convert.** The *Convert file formats* dialog appears on the screen. Select the type of conversion, the source file and destination file.

17.7 Language

Management
`Language...`

Select the menu command **Management ▸ Language.** The *Select STEP 5/ST language* job box appears on the screen. Enter an X beside the required language and click **Enter**.

As an option, you can have the language selection box displayed each time you restart STEP 5/ST.

17.8 Colors

Management
`Colors`

Select the menu command **Management ▸ Colors.** The S5COLOR Screen colors job box is displayed.

Black and White Display for STEP 5/ST V 7.0

STEP 5/ST is designed for a color monitor.

If you connect a monochrome monitor to your PC, the dialogs are displayed in gray tones. If you prefer a black and white display, you can activate this option for your work station by copying the MONO@@FT.DAT file to your home directory and renaming it @@@@@@FT.DAT.

The MONO@@FT.DAT file is in the \S5_INST subdirectory in the system directory.

If you select the black and white display, this affects STEP 5/ST optional packages and COM packages as well as tools such as S5DRV.EXE on your work station.

Black and white display has priority over user-specific color scheme.

To deactivate the black and white display at your work station, remove the @@@@@@FT.DAT from your Home directory (see also search order).

User-Specific Color Display for STEP 5/ST V 7.0

You can change the color scheme for STEP 5/ST.

This is particularly useful if you want to improve the gray tone display on a monochrome monitor or cannot distinguish certain colors due to the color setting of the monitor.

The user-specific color setting is selected using the menu item Management/ Color setting and is stored in the S5@@@@FT.DAT file in the Home directory

The user-specific color setting only affects STEP 5/ST on your workstation.

COM packages and tools such as S5DRV.EXE are displayed in the standard colors.

To deactivate the user-specific color display on your workstation, remove the S5@@@@FT.DAT file from your home directory.

Documentation

18

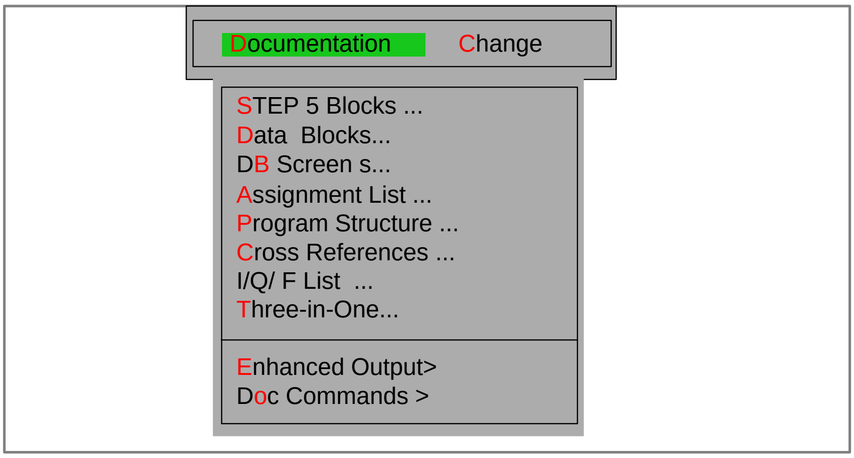
Overview

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, for example:

- 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, for example:

- output the cross reference list according to selected operands
- output the assignment list sorted according to symbolic operands



Chapter Overview

18.1 Overview of the Documentation Functions

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 (see Section 18.2).

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 (see Section 18.3).

Doc Command for Enhanced Output

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 (see Section 18.5.6).

Hardcopy

You can print a hardcopy as follows:

1. with the ***SHIFT + PRINT*** key

   Under Windows 95, this key combination creates a “snapshot” of the screen that you can print using system functions.

18.2 Standard Output

Menu

Figure 18-1 shows the menu options for 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.

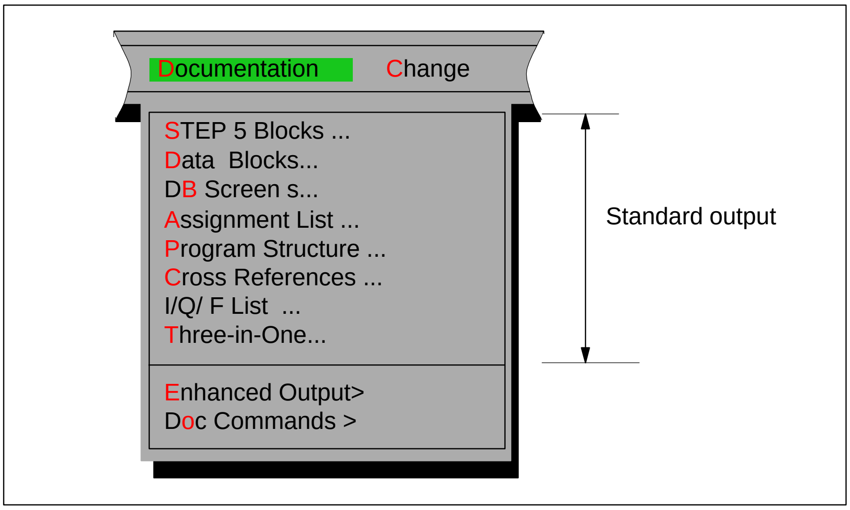


Figure 18-1 Menu Options for Standard Output

Note

For standard output, no cross reference list (file *XR.INI) is necessary.

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
```

Figure 18-2 Printout of a STEP 5 Block

Settings

Please check if the following is set::

- Program file
- method of representation STL, LAD or CSF
- Footer file (only if *footer: yes* is set)
- Symbols file (only if *symbols:yes* is set)
- XRF file (→ *Make XRF*)
- Mode (onli`rep.`ne when you want to output from the PLC)
- Printer file (the defaults `NONAMELS.INI` apply to the PT88)
- With or without comment

For more information about settings, refer to the project settings, *Section 4.1.1*.

Activating the Functions

Select a menu command, for example **Documentation ▸ STEP5 Blocks...**

A job box, in this case, Print *STEP 5 blocks,* is displayed. In this box, you can browse and make your selections (see *Section 3.6)*

Output

You can output to a file, to the screen or to a printer.

If the screen display covers more than one page, the prompt `Continue? yes/no` is displayed. You can clear this box from the screen with the space bar.

As an option, you can add a customized footer to the printout.

18.2.1 STEP 5 Blocks

Function

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.

Documentation
`STEP 5 Blocks`

Select the menu command **Documentation ▸ STEP 5 Blocks**. The *Print STEP 5 block(s)* job box is displayed. Here, you can make your selections.

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.
`wit STL addresses`	Only when STL is selected: select the type of address information.

18.2.2 Data Blocks

Function

With this function, you can either output individual or all the data blocks of a program.

Documentation
`Data Blocks`

Select the menu command **Documentation ▸ Data Blocks**. The *Print data blocks* job box is displayed. Here, you can make your selections.

Example of an output

With comments was selected in the *settings* (see Section 4.1.1, Blocks tab).

```
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 18-3 Example of Data Block Output

18.2.3 DB Screens

Function

With this function, you can output data blocks containing screen forms.

Documentation
`DB Screens`

Select the menu command **Documentation ▸ DB Screen Forms**. The *Output DB screens* job box is displayed. Here, you can make your selections.

18.2.4 Assignment List

Function

With this function, you output an assignment list to printer or file.

Documentation
`Assignment List`

Select the menu command **Documentation ▸ Assignment List**. The job box *Print assignment list* is displayed. Here, you can make your selections.

Example

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 18-4 Example: Output of an Assignment List

18.2.5 Program Structure

Function

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.

Documentation
`Program Structure`

Select the menu command **Documentation ▸ Program Structure**. The *Output program structure* job box is displayed.

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 18-5 Program Overview with DB

Block Call IDs

The blocks are output with call IDs. These show you the type of call in the program.

ID	Explanation
–	Block is called unconditionally
=	Block is called conditionally
#	Block call follows a DO 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

18.2.6 Cross References

Function

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.
  cross references to data
  cross references to I/Os
  cross references to block calls

  (S flags are not output in the cross reference list**.**)
- cross references to individual symbolic or absolute operands (e.g. –MOTOR, I1.0)

The cross references are sorted by absolute operands. An entry contains the following:

- the operand
- the symbol
- the block and segment
- an identifier showing use (see Figure 18-6).

Documentation
`Cross References`

Select the menu command **Documentation ▸ Cross References**. After you call the function, the *Output XREF list* job box is displayed. In the following table only the inputs specific to this function are explained.

Input field	Explanation
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.

```
X 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*
```

Operand
Symbolic name
Block
Block no.
Segment no.
Operand as scan
Operand as assignment

Figure 18-6 Example of a Cross-Reference List

Meaning of the IDs

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.

18.2.7 I/Q/F List

Function

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, Q, F, operand areas. One line is reserved for every two bytes of an operand area, in which the 8 possible bits are marked (see Figure 18-7).

- a byte (**B**)
- a word (**W**)
- a double word (**D**)

Documentation
`I/Q/F List`

Select the menu command **Documentation** ▸ **I/Q/F list**. The *Output I/Q/F list* job box is displayed. Here, you can make your selections

Example

```
                                                              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

Existing 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 18-7   Example of a Standard I/Q/F List

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.

18.2.8 Three-in-One

Function

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. For standard output no cross-reference list (file *XR.INI) is necessary.

Documentation
`Three-in-One`

Select the menu command **Documentation ▸ Three-in-One**. The *Output three-in-one job* job box is displayed.

18.3 Enhanced Output

Overview

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**.

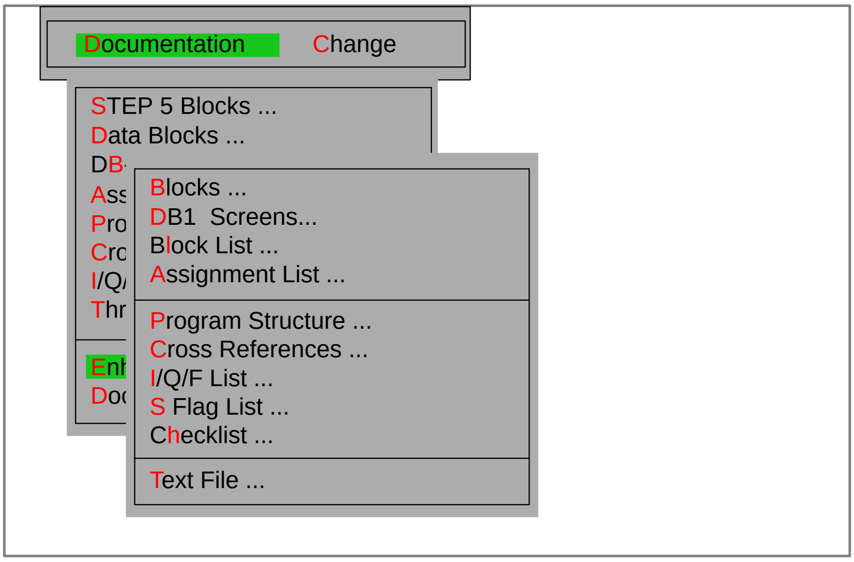
Output is also possible with continuous lines (see Figure 18-8 and Figure 18-9). You can print on either A3 or A4 paper. The printout on A4 paper is a compressed form of the A3 printout. The objects to be documented must be located on diskette or hard disk. If you only have programs in the PLC memory, 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 (see Section 18.4)* 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 select the printer setting in the → *Editor, Printer Parameters* before printing out.

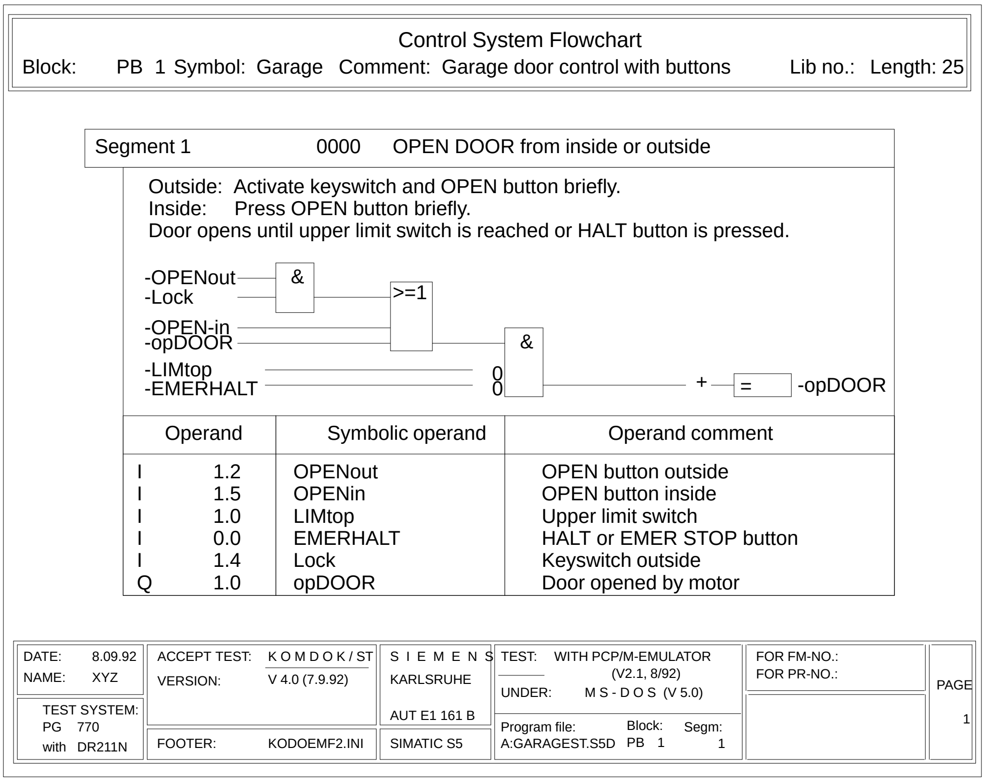
Selecting Enhanced Functions

When you select the *enhanced output* function, a menu is displayed in which you can select the following elements for output:



Example of a Printout

The first printout (Figure 18-8) illustrates enhanced output and the second (Figure 18-9) is a standard printout. Note the difference between the two figures.


Control System Flowchart

Block: PB 1 Symbol: Garage Comment: Garage door control with buttons Lib no.: Length: 25

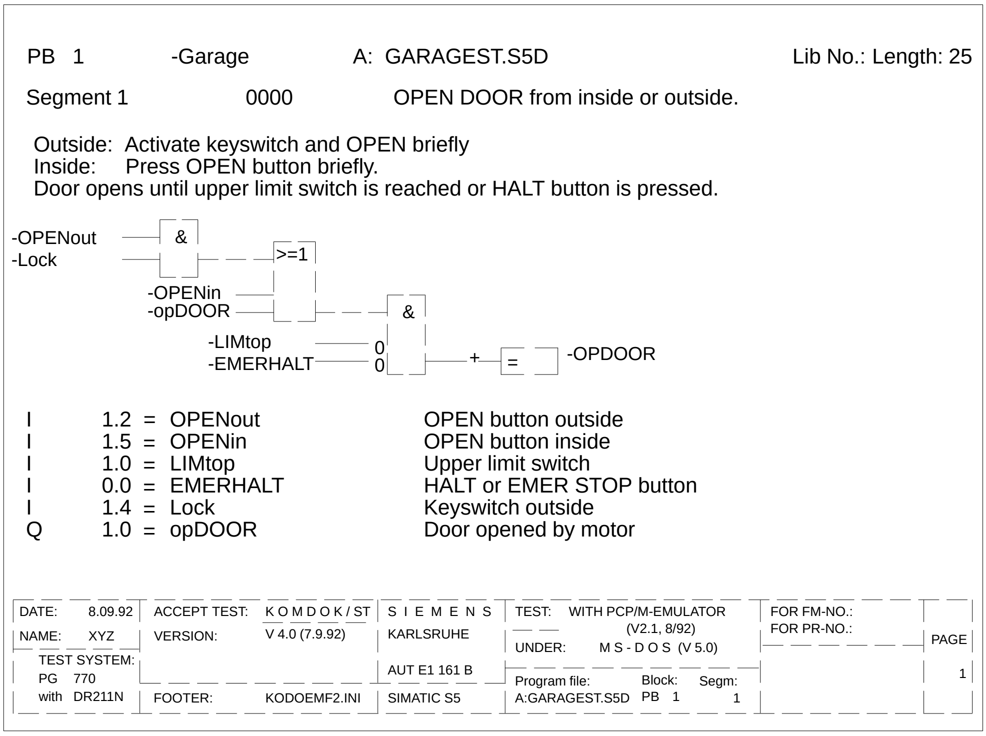
Segment 1 0000 OPEN DOOR from inside or outside

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: K O M D O K / ST VERSION: V 4.0 (7.9.92)	S I E M E N S KARLSRUHE	TEST: WITH PCP/M-EMULATOR (V2.1, 8/92) UNDER: M S - D O S (V 5.0)	FOR FM-NO.: FOR PR-NO.:	PAGE
TEST SYSTEM: PG 770 with DR211N	FOOTER: KODOEMF2.INI	AUT E1 161 B SIMATIC S5	Program file: A:GARAGEST.S5D Block: PB 1 Segm: 1		1

Figure 18-8 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.

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: K O M D O K / ST VERSION: V 4.0 (7.9.92)	S I E M E N S KARLSRUHE	TEST: WITH PCP/M-EMULATOR (V2.1, 8/92) UNDER: M S - D O S (V 5.0)	FOR FM-NO.: FOR PR-NO.:	PAGE
TEST SYSTEM: PG 770 with DR211N	FOOTER: KODOEMF2.INI	AUT E1 161 B SIMATIC S5	Program file: A:GARAGEST.S5D Block: PB 1 Segm: 1		1

Figure 18-9 Simple Printout of a Control System Flowchart

18.3.1 Output KOMDOK Blocks

Function

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).

Documentation
`Enhanced Output`
`Blocks`

Select the menu command **Documentation ▸ Enhanced Output ▸ Blocks**. The job box *Output KOMDOK blocks* is displayed.

In the following table only the inputs specific to this function are explained.

Input	Explanations
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.
Layout	If you press ***F7*** an example of a standard format is displayed.
Update XRF	The XRF file is updated before the blocks are output.

18.3.2 KOMDOK DB1 Screens

Function

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).

Documentation
`Enhanced Output`
`DB1 Screens`

Select the menu command **Documentation ▸ Enhanced Output ▸DB1 Screens**. The job box *KOMDOK output DB1 screens* is displayed.

18.3.3 KOMDOK Block List

Function

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

Documentation
```
Enhanced Output
   Block List
```

Select the menu command **Documentation ▸ Enhanced Output ▸ Block List**. 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.

18.3.4 KOMDOK Assignment List

Function

You can output an assignment list as follows:

- in sequential form, as edited
- sorted according to absolute operands
- sorted according to symbolic operands.

Documentation
```
Enhanced Output
  Assignment List
```

Select the menu command **Documentation▸ Enhanced Output ▸ Assignment List**. The *Output KOMDOK assignment list* job box is displayed.

You can output the assignment list in the following modes:

Inputs	Explanation
Unsorted	Unsorted output. The symbols setting is not relevant.
Sorted by absolute operands	The output is sorted by 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.
Sorted by symbolic operands	The output is sorted by 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.

When you exit the job box with ***Output***, the following message flashes up:

```
Printout assignment list
```

If this message disappears, the function is completed and, providing no error message has occurred, the assignment list is output.

18.3.5 KOMDOK Program Structure

Function

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* in the project settings, *Section 4.1.1*).
- The maximum nesting depth that can be recorded is 9.
- You can output with or without data blocks.

The following calls are listed:

Call	Explanation
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

Documentation

```
Enhanced Output
  Program
  Structure
```

Select the menu command **Documentation ▸ Enhanced Output ▸ Program Structure**. The *Output KOMDOK program structure* job box is displayed. The fields specific to this function are explained below.

Input field	Explanations
Program file	Cannot be selected here. Must be preset (*project settings, Section 4.1.1*)
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 to	As in all job boxes.

When you exit the job box with ***Output***, the following message flashes up:

```
Printout program structure
```

When this message disappears and if no error message has occurred, the function is completed and the program structure output.

Example of a Printout

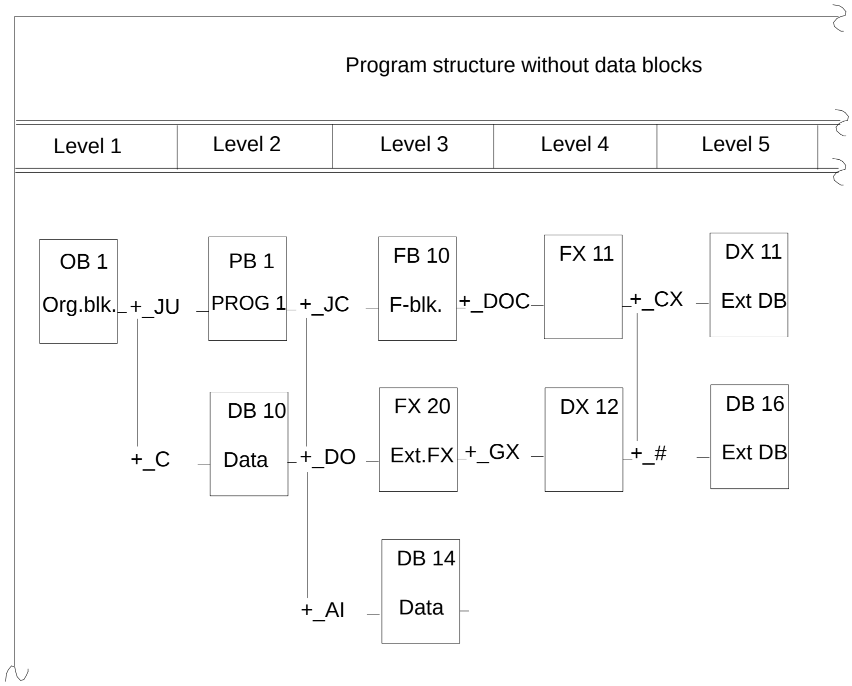


Figure 18-10 Output of a Program Structure without DB

18.3.6 KOMDOK Cross Reference List (XRF List)

Function

With this function, you output cross references within the program file according to certain criteria from an existing cross reference list (*XR.INI).

The following information is provided:

- cross reference list according to operand IDs, for example 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, Make XRF*).

If you modify the program, the cross reference list must be regenerated.

Documentation
```
Enhanced Output
  Cross
  References
```

Select the menu command **Documentation ▸ Enhanced Output ▸ Cross References**. The job box *Output KOMDOK XREF list* is displayed. The input fields specific to the function are explained below.

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, S 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.

When you exit the job box with ***Output***, the following message flashes up:

```
Printout XRF list
```

When this message disappears and if no error message has occurred, the cross reference list is output.

18.3.7 KOMDOK I/Q/F List

Function

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**)

Documentation
```
Enhanced Output
I/Q/F List
```

Select the menu command **Documentation ▸ Enhanced Output ▸ I/Q/F list**. An I/Q/F list is printed out or output to a file. 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, Make XRF*).

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.
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 I/Q/F List of the Inputs

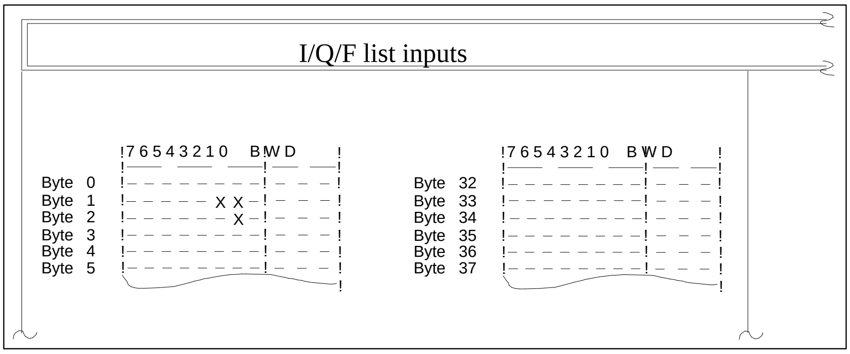


Figure 18-11 I/Q/F List of the Inputs

18.3.8 KOMDOK I/Q/F List for S Flags

Documentation
`Enhanced Output`
`S Flag List`

This function outputs the I/Q/F list for the S flags (see Figure 18-11 I/Q/F list).

18.3.9 KOMDOK Checklist

Function

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.

Documentation
```
Enhanced Output
  Checklist
```

Select the menu command **Documentation** ▸ **Enhanced Output** ▸**Checklist**. The job box *Output KOMDOK checklist* is displayed.

18.3.10 KOMDOK Text Files

Function

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.

Documentation
```
Enhanced Output
  Text File
```

Select the menu command **Documentation** ▸ **Enhanced Output** ▸ **Text File**. The job box *Output KOMDOK text file* is displayed.

18.4 Doc Commands

Overview

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:

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 (Figure 18-12). 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*

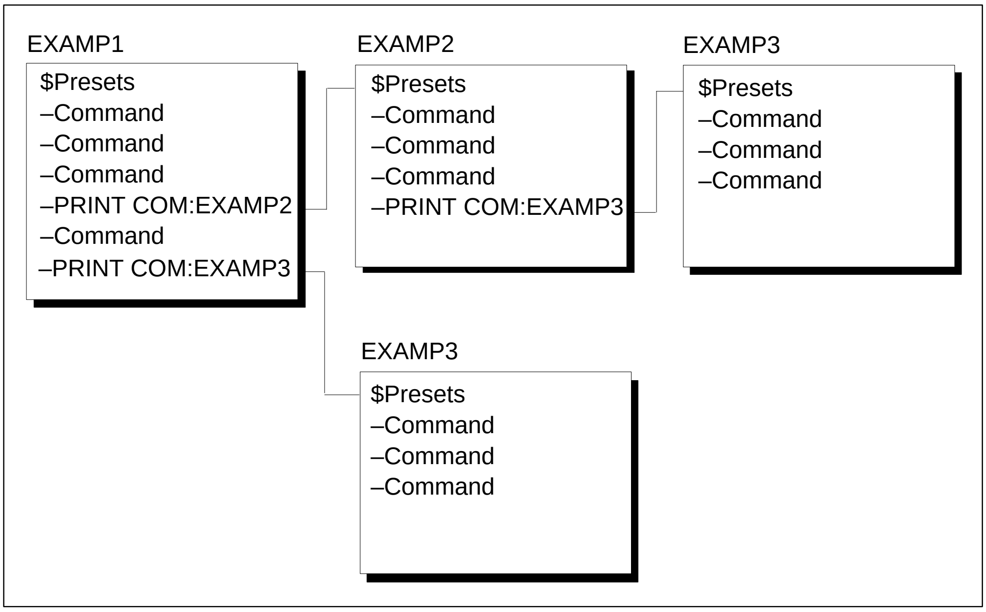


Figure 18-12 Structures of the Doc Commands (Example)

18.4.1 Presets

Table 18-1 Doc Command for Presets

Doc Commands	Explanation
`$LAD, $CSF, $STL`	Method of representation**:** of the Ladder Diagram (LAD), Control System Flowchart (CSF), Statement List (STL).
`$KAT:X:\ ...\`	Create a directory: the set directory is used with the SUBMIT commands $PROG, $SYMB, $FOOT, $DLST and -DOCCOMM (see Table 18-5).
`$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`	The printer file is identified by this name. The program searches first in the ...\S5_HOME catalog (printer files created or changed by the user) and then in the ...\S5_SYS\DR_INI\ catalog (printer files originally supplied and copied to this directory by the installation program).
`$PATH:X:NNNNNN (PATH name)`	Path name: has no effect.
`$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.
`$CONTENT`	Directory: from this doc command onwards, a directory is kept. This preset can no longer be reset in the active submit.
`$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.

The commands $PROG, $SYMB, $FOOT. $PLST and –DOCCOMM must identify the entire directory. This can be achieved in three ways:

1. A $KAT command is specified to set the directory and only the drive and file name are specified in the SUBMIT command, for example:
   `$KAT:C:\DATEN\TEST`
   `$PROG:C:NONAME`
   In the SUBMIT, `C:\DATEN\TEST\NONAMEST.S5D` is used as the program file.

Table 18-1 Doc Command for Presets

Doc Commands	Explanation

2. A $KAT command is used and only the drive and the file name are specified, for example:
   `$PROG:C:NONAME`
   The directory is the directory for the particular file type from the project settings, in this case the program file.

3. The full directory is written in the SUBMIT command, for example:
   $PROG:C:\DATEN\TEST\NONAME

18.4.2 Commands

Table 18-2 Doc Commands for Blocks

Doc Commands	Explanation
–BLOCK:A	All blocks
–BLOCK:#	All documentation files
–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.

Table 18-3 Doc Commands for Blocks with Options

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).

Table 18-4 Doc Command for Block List

Doc Commands	Explanation
–BLST	Output the block list of the preset program file.

Table 18-5 Nested Doc Commands

Doc Commands	Explanation
`–DOC–COM:x:nnnnnn`	The doc command file nnnnnnSU.INI is called and started.

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*).

Table 18-6 Directory

Doc Commands	Explanation
`–CONTENT`	The directory is output with the current footer. The page number begins automatically at I and is restored on completion of the directory.
`–CONTENT:n`	The page numbering of the specified directory begins at n (n= 1, 2...)

A directory of all previous printouts is output if you activate the default *$CONTENT* (see *Table 18-1).*

Table 18-7 Check list

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

Table 18-8 Program structure

Doc Commands	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

Table 18-9 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.

Table 18-9 XRF list

Doc Commands	Explanation
`–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**.

Table 18-10 I/Q/F List

Doc Commands	Explanation
`–XRF:IQF`	Output the I/Q/F list.
`–XRF:IQF S FLAGS`	Output the I/Q/F list of the S flags**.**

Table 18-11 Assignment List

Doc Commands	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).

18.5 Editing Doc Commands

Overview

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***.

Documentation
```
Doc Commands
  Edit
```

Select the menu command **Documentation ▸ Doc Commands ▸ Edit**. A job box is displayed in which you select a submit file. The new file name is entered in the settings box (→ *Project, Settings, Section 4.1.1*). As soon as the screen below is displayed, the cursor is positioned in the first editing line. You can now edit.

```
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    •
        •

F Text1  F Text2     F Text3     F Text4     F Text5   F Text6     F Text7  F Help
1 Field  2 Fetch Fld 3 Fetch Fil 4 Fetch Lin 5 Extras  6 Fct Keys  7 Enter  8 Cancel
```

Figure 18-13 Submit File Editor

18.5.1 Function Key Assignment

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.

Table 18-12 Function Key Assignment

Key level 1	2	3	Effect of the Function Keys
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.
		F3	**Select** Select file dialog box is opened
		F6	**Enter** The field is stored in the selected file.
		F8	**Cancel** Return to previous key level without action.
		SHIFT F8	**Help**
	F4		**Delete**
	F5		**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.
		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	**Cancel** Return to previous key level without action.
		SHIFT F8	**Help**
	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
		SHIFT F8	**Help**
	F8		**Cancel** Return to previous key level without action.

Table 18-12 Function Key Assignment, continued

1	2	3	
	SHIFT F8		**Help**
F2			**Fetch Fld** The currently buffered field is fetched and inserted after the cursor position.

Table 18-13 Existing Submit File

Key level			Effect of the Function Keys
1	**2**	**3**	
F3			**Fetch Fil** 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.
	F3		**Select** Select file dialog box is opened
	F6		**Enter** The file is fetched without function assignment (as with ***F1***).
	F8		**Cancel** Return to previous key level without action.
	SHIFT F8		**Help**

Table 18-14 Fetch Line/Find Text

Key level			Effect of the Function Keys
1	**2**	**3**	
F4			**Fetch lin** A previously deleted line (with the delete key) is fetched back → *Key assignment.*
F5			**Extras**
	F4		**Del LINE**
	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	**Cancel** Return to previous key level without action.

Table 18-15 Replace Character String

Key level 2	3	4	Effect of the Function Keys
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	**Cancel** 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.
		F2	**No** The text is not replaced.
		F8	**Cancel** 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		**Cancel** Return to previous key level without action.
	SHIFT F8		**Help**

Table 18-16 Jump

Key level 2	3	4	Effect of the Function Keys
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.
	SHIFT F8		**Help**

Table 18-17 Editing Function Keys

Key level 1	2	3	Effect of the Function Keys
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 Lin** Fetch back the characters deleted with the delete key (→ *Key assignment*).
	F6		**Enter** The function key assignment is entered.
	F8		**Cancel** The function key assignment is entered.
	SHIFT F8		**Help**

18.5.2 KOMDOK Test Doc Commands

Function

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.

Documentation
`Doc Commands`
`Test`

Select the menu command **Documentation ▸ Doc Command ▸ Test**. The *Test doc commands* job box is displayed. Here, you enter the name of the file you want to check. When you click ***Test***, the test is started and the results are displayed.

Displaying the Error List

Errors found while the **Doc Command ▸ Test** function is running are saved in an error file. You can output these files with **Doc Command ▸ Output Log File.**

Note

If no errors are found, no error file is created.

Error Message

```
            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
```

Figure 18-14 Error Message

18.5.3 Output Log File

Function

Errors found in the functions *Test doc commands* or *Execute doc command* are written to a log file that you can output with this function.

Documentation

```
Doc Commands
  Output Log
  File
```

Select the menu command **Documentation ▸ Doc Command ▸ Output-Log File**. The *Output log file* job box is displayed. Here, you can make your selections. The name of the generated error file is set here.

Fields	Explanation
Log file	Name of the error log file. The generated file name is the default. You can select a different name with ***F3***.
Screen	Output directly on the screen.
Printer	Output directly to the printer according to the selections made for printer parameters.
File	Output to a selectable file.

18.5.4 Run Doc Command

Function

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.

Documentation

```
Doc Commands
  Run
```

Select the menu command **Documentation ▸ Doc Command ▸Run**. The job box *Run doc commands* is displayed. Here, you enter the name of the file whose doc commands you want to use in the *Doc command file* field. You can select a file by pressing ***F3***. Once you start the function with ***Execute*** the doc commands are processed.

Note

If errors occur, you can branch to an error list.

18.5.5 Output Doc Command

Function

You can print out the content of a doc command file.

Documentation

```
Doc Commands
  Output
```

Select the menu command **Documentation ▸ Doc Command ▸ Output**. The job box *Output doc command file* is displayed. Here, you enter the name of the file you want to print in the *Doc command file* field:. You can select a file with ***F3***. When you click on ***Output*** the doc commands are printed out.

18.5.6 Edit Doc Command Structure

Function

Within the doc commands you can include statements (→ *Edit structure*) 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.

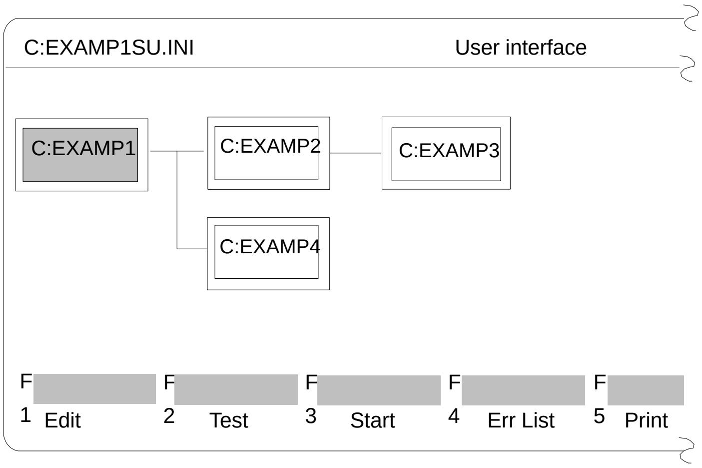


Figure 18-15 Interconnecting Doc Command Files

Documentation
Doc Commands Edit Structure

Select the menu command **Documentation ▸ Doc Command ▸ Edit Structure**. 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 ***Edit*** 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, 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 following table shows which key combinations are possible and the effects of the function keys.

Table 18-18 Function Key Assignment

Key level 1	Key level 2	Effect of the Function Keys
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		Err List The error list of the doc command file marked on the screen is displayed and, if required, printed out.
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		**Find** 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**
F8		**Return** Return to the calling level without action.
SHIFT F8		**Help**

18.5.7 Output Doc Command Structure

Function

The structure of connected doc command files is printed out in A3 or A4 format or output to a file (LS.INI).

Documentation
Doc Commands
Output
Structure

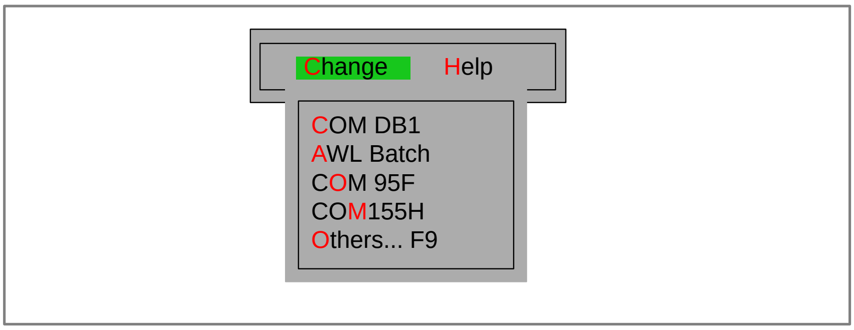
Select the menu command **Documentation ▸ Doc Command ▸ Output Structure**. The job box *Output doc command structure* is displayed.

Input field	Explanation
Doc command file	Name of the doc command file about which you want to see structure information. Starting from this file.
Structure with doc commands	The content of the doc command files involved is also printed out, each file on a separate page.

19 Change

Overview

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.



Change
`Others F9`

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 *Others* 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 SIMATIC 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.

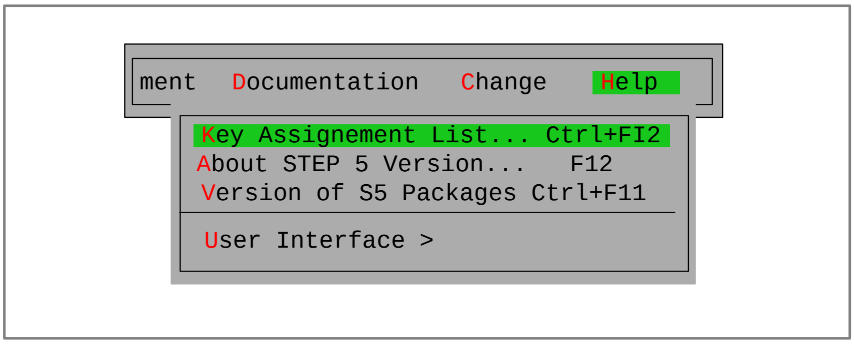
You make your selection in this box (→ *Graphical user interface*). Once you have selected a package and confirmed your selection with ***Start***, the user interface of the selected package is displayed.

20 Help

Overview

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*** to ***F10*** and ***SHIFT F1*** to ***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.



Chapter Overview

20.1 Key Assignment List

Help
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.

20.2 About STEP 5/ST Version

Help
About STEP 5/ST Version

A box is displayed containing information about the currently active STEP 5 package.

20.3 Version of S5 Packages

Help
Version of S5 Packages

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 20-1 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 Section 4.1

Operation

The job box for the version of the S5 packages is displayed. Here, you make your selections (→ *User interface, Job box*).

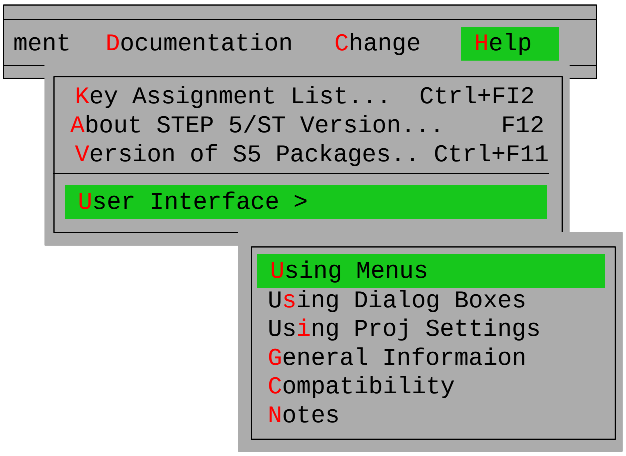
Directory

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.

20.4 User Interface

Help
User Interface >

This function provides you with a description of the ways in which you can find information in the help system.



Part 4:
Other Simatic S5 Programs

STL Editor/Batch Compiler 21

Overview

The STL editor/batch compiler optional package has an independent editor for programming in the STL mode of representation and an independent compiler for compiling statement lists into a runnable STEP 5 program.

With the batch compiler you can decompile from a STEP 5 program so that, for example, modifications made to the tested program can be entered in your source files and your statement list can be updated.

Chapter Overview

The editor also tests the compiled STEP 5 program for your specific PLC and creates an error list.

21.1 How the the STL Editor/Batch Compiler Functions

Overview

Creating a STEP 5 program with the STL editor/batch compiler differs in the following way from the LAD, CSF, STL package:

- In the LAD, CSF, STL package the statement list is directly edited in the program file and immediately compiled into machine code.
- In the STL editor/batch compiler package editing and compilation are separate processes.

Editing

During the first step (editing), you write a sequential text file (STL source file) with the STL editor. It can contain a statement list, which has been created exclusively with symbols.

Saving

When data is saved using the *Enter* function or the **Insert** key, the package automatically creates an intermediate file in addition to the STL source file. This intermediate file contains a code which is independent from national languages (language-independent), but is not yet a machine code. During this first *compilation* your statement list is checked for syntax and format.

Compilation

During the second step you start the compilation with a function key. Here the batch compiler converts the intermediate file into a STEP 5 program file. If you programmed your statement list symbolically, the batch compiler requires a symbol file with the relevant assignments.

Test

During the compilation of the program file, the assignments are tested. If you have entered a specific PLC type, the system checks whether the operations contained in the program are permitted for your destination PLC (*PLC specific test*). A program file created with the STL editor/batch compiler is identical to a program file created in an LAD, CSF, STL package.

Decompilation

You can also create a source file from a program file with the STL editor/batch compiler. This may be necessary, for example, after the STEP 5 program has been tested on the PLC and debugged. It does not matter whether the program was edited in the LAD, CSF, STL package or in the STL editor/batch compiler package. During such a decompilation the batch compiler first generates an intermediate file from the program file. The STL source file is then created from this intermediate file. The STL source file can also be created directly in one step from a program file.

Test Run

Tests are carried out during the compilation. In addition to these tests, a test run for the program file blocks is carried out. During the test run a check is made, whether formal operands and actual operands in function blocks have been correctly assigned and match. All errors are listed in an error list and can be printed out.

The error list, however, only contains the errors of the last session and is overwritten after every new compilation or test. It is therefore advisable to print out the error list each time it is created. If a phase is completed without any errors, no error list is created and an existing list is deleted.

21.2 Creating STEP 5 Blocks

Overview

In the STL editor, you can create your control program as a statement list with the advantages of a text editor. You write your statement list using the same commands and the same syntax as in the LAD, CSF, STL package. The only difference is that you must keep to certain conventions such as control characters for the start of blocks and comments.

All block types that can be created in the LAD, CSF, STL package can also be created with the STL editor/batch compiler package. All comments, that can be written in a block, can also be created. However, plant comments (DOC file) cannot be written. The STL editor also permits additional comments that can be placed anywhere in the statement list. However, these comments are not transferred to the program file and are lost during a decompilation into the same STL source.

DB0 (reserved in the PLC for the block address list), DB1 (for the I/O assignment of the S5-135U and S5-155U), DX0 (for the system parameter assignment, CPU 928, R-processor, S5-155U), DB2 (for the controller list of R64), GRAPH 5 and assembler blocks are not possible.

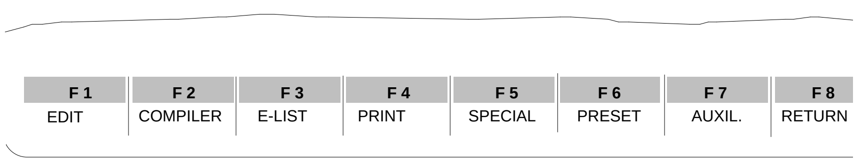
Presetting

In this dialog, you select four files:

1. The STL source file you want to edit (A0.SEQ).
2. The intermediate file which is created when you save with the Enter key or Insert key and contains the statement list compiled into the intermediate code (A1.SEQ).
3. The symbols file, which contains an assignment list (Z0.INI).
4. The program file in which the STEP 5 program will be written after the compilation (ST.S5D).

These four files are automatically entered using the same name and can be altered if necessary. The STL source file and the intermediate file always have the same name.

The STL editor/batch compiler package provides the following functions:



Function Key	Function
EDIT	For creating and editing the STL source file
COMPILER	For compiling and decompiling
E-LIST	For the error list of the test runs
PRINT	For printing the STL source file

Function Key	Function
SPECIAL	Functions, for generating intermediate files and the STL source file
PRESET	For changing the presetting;
AUXIL.	Functions, for administrating blocks in the current program file and
RETURN	For exiting the STL editor/batch compiler.

21.2.1 Editing Functions

Overview

The STL editor displays an editing dialog box on the screen, which is ready for a statement list. The editing dialog box consists of the following:

- A header containing the name of the STL source file.
- The input fields for ADDRESS, STATEMENT, OPERAND SYMBOL and STATEMENT COMMENT, arranged in columns.
- The menu with the editing functions.

Function

The editor provides several functions so that your program can be edited more easily. They are comparable to the symbol editor functions:

F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8
BUFFER	COPY	DELETE	SEARCH	REPLACE	ENTER	SAVE	MODE

With the the BUFFER and COPY functions, any character sequence or text block can be written to a buffer or to a sequential file and copied at any point in the program (copy and paste function). That means, you can move or include segments again and again.

You can also read in other STL source files or single blocks from them with the copy function. The buffer, copy and delete functions can be combined with a repetition factor.

The SEARCH and REPLACE functions make it easier to correct your program:

- You can move quickly and to locations in your file. You can alter individual character sequences, for example symbols or operands, in your entire statement list using only one function. You can select either the insert or overwrite mode.
- With SAVE you can save your file without exiting the editor. Saving your STL source file with this function is a sensible precaution if you have to interrupt a programming session. The ENTER function saves your file and at the same time compiles it into the intermediate file and then exits the editor.

Control Characters

To make compilation of the STL source file into a STEP 5 file possible, certain control characters and conventions must be observed when editing.

#TY identifies the PLC type. After this control character you can enter the PLC, on which the program will run. The entry must match the setting in the language category field of the presetting dialog. During compilation into the program file, the batch compiler checks whether the edited operations are permitted in the operations set of the PLC. The PLC type can be entered at the beginning of the STL source file and at block boundaries.

The following names are permitted for the PLC language category.

PLC Type	Processor	Language Category Names
S5-100 U	CPU 100 CPU 102 CPU 103	CPU100 CPU102 CPU103
S5-101 U		S5-101U
S5-110 S		S5-110S
S5-115 U	CPU 941 CPU 942, 943, 944	CPU 941 CPU 942
S5-130WB		S5-130 W
S5-135 U	CPU 921 CPU 922 CPU 928 CPU 928B	CPU 921 CPU 922 CPU 928 CPU 928B
S5-135W		S5-135 W
S5-135 WB		S5-135B
S5-150 A/K		S5-150A
S5-150 S/U		S5-150S
S5-155U	CPU 946/947	S5-155 U
I/O processor	IP 257	IP 257

The program file is only compiled if the name of the PLC in the presetting (”LANG CAT AREA” field) matches the entry in the #TY lines of the STL source file. If they do not match, the compilation is canceled at the #TY line. If you enter “NO” in the presetting for the language category, the program is compiled without a PLC specific test run.

Include Command

The #I include command allows linking of any file and can be entered in the STL source file at the beginning of the file or at block boundaries, in other words after BE. The command #I is followed by the file name. Here, it is important that the drive is also entered (for example: #I B:TEST).

The file is included at the intermediate code level, i.e. the file to be included must exist as an intermediate file. If the disk drive is not entered, the drive specified for the intermediate file in the presettings is accessed.

If blocks with identical names are found in both files, they must be renamed before compilation. You can avoid this problem, however, by assigning symbolic names to the blocks in your STL source files.

If you use symbolic names, the corresponding symbol file must exist since you cannot compile without the block type and number.

The include command is particularly suitable for user-defined libraries: standardized programs can be modified with include files to adapt them to a particular task. To make a modification, for example you then only need to swap the include files. After the modification the latest version is used to create the program file.

21.2.2 Compilation

Creating a Program File

The batch compiler can compile all blocks, a group of blocks or an individual block from the intermediate file or the STL source file into the program file. If you have saved all the modifications in your statement list in the source file with ENTER, only the intermediate file needs to be compiled. If not, you must start compilation of the STL source file, which automatically creates an updated intermediate file.

If you programmed your STL source file symbolically the preset symbol file is linked to the intermediate file during the compilation into the program file. A symbol file is not created by the STL editor, but must be created with the symbols editor. If another file is included with the #I include command, make sure that the symbols file also contains the symbols for this file.

In the command lines of the compiler, you can specify various options: whether machine code should be generated or the program simply tested for errors, and whether you want to confirm before overwriting blocks. You can also decide to have the compiled program printed out.

Decompilation from a Program File

Neither STL source files nor intermediate files exist for blocks which were written with the LAD, CSF, STL package. The STL editor/batch compiler can create these files from a program file.

The intermediate file is created after decompiling a block, a block group or all blocks from a program file. When a block, a block group or all blocks are decompiled from the program file, you can first create the intermediate file or create the STL source file immediately. You can then modify or extend the source file.

When you decompile a program, you decide what your “new” STL source file will look like. The statements contain either symbols only or absolute parameters only, or both. The control character for the language category identification (PLC type) is also entered in the intermediate file if a category was selected in the presettings.

The STL editor can process files with up to 65535 lines. The number of lines of the STL source file, however, not only depends on the number of STEP 5 statements, but also on special statements, comment lines etc. If the program file you want to decompile is too long, the blocks must be be distributed in several intermediate files.

Standard function blocks as well as Graph 5 and assembler blocks are not decompiled.

Compilation Checks

During the compilation/decompilation, the intermediate code is checked to make sure that the resulting statement is permitted. It is also checked to make sure that it is permitted in terms of the block type. The language category is checked for the PLC type entered in the presettings. If you program using symbols, the assignments to the operands are checked.

If you have specified both an absolute as well as a symbolic operand in the STL source file, the symbols file is also checked to make sure that it matches. If the parameters do not match, the absolute parameter assigned to the symbol from the symbol file is used and a warning is entered in the error list. If you program with absolute operands, the symbols file is irrelevant. Errors found during these checks are entered in the error list.

21.2.3 Printing

Overview

You can create a listing of the STL source file using the print function (in function selection). However, this function only outputs the preset STL source file to the printer.

In the command lines of the compiler function, you can select a print function, so that you can record the result of every compilation run including the test run.

Layout

The STL editor/batch compiler provides the printing formats, which are normally used in the STEP 5 basic package for the layout of your printer output. You can choose between standard output, normal print, condensed print and super condensed print. The footer must be 132 characters wide for the A3 format (F2.INI file) and 80 characters wide for the A4 format (F1.INI file). The symbol comment is also displayed when super condensed print is selected.

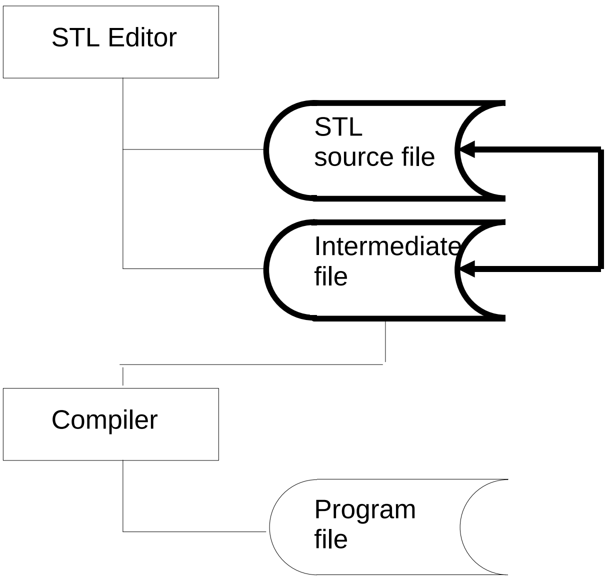
21.3 The A1.SEQ Intermediate File

The intermediate file is the central file in the STL editor/batch compiler package. It is the basis for all compilation as it is language independent and not yet in MC5 machine code. At any time, you can create the following from the intermediate file:

- STEP 5 program files,
- STL source files,
- program versions for a specific process/plant
- foreign language program versions

For this reason, it is essential to save the intermediate file and it is advisable to exit an STL source file with *ENTER* so that the intermediate file is updated.

21.3.1 Relationship between the STL Source File and the Intermediate File



The STL source file and the intermediate file are closely connected: they have the same name and their file identification differs only in one character (A0.SEQ,A1SEQ). The name of an intermediate file can never be altered separately from the STL source file. The two files can, however, be on different drives. Using identical names guarantees that your edited programs are compiled in the correct intermediate file when you enter your source file.

Decompilation

When you decompile a program, the program is also stored in the STL source file with the same name. Bear this in mind if the STL source file and the intermediate file are not in the same version or if you do not want to overwrite the "old" statement list, for example if your first STL source file contains additional comments. Comments are not written back to the program file and are lost after the decompilation.

The following rules apply when recreating an STL source file from an intermediate file:

- If no STL source file exists, it is created automatically when you click Edit. Its name is the name entered in the presettings.
- If an STL source file with the same name exists, the intermediate file must be decompiled explicitly into the STL source file using the **INT>SEQ** special function (see below). The STL source file is then overwritten.
- If you want to keep an "old" STL source file, enter a name for the new STL source file before the *decompilation* of the program file. The intermediate file is then decompiled into this new file.

You should always check the files selected in the presettings before compiling and especially before decompiling.

Note

The SEQ>MC5 and MC5>SEQ functions automatically create an updated intermediate file.

21.3.2 Special Functions

F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8
SEQ>INT	INT>SEQ	SEQDELETE	INTDELETE	COPY	TEST RUN	SYM-GEN	RETURN

The special functions offer various conversions for creating STL source files and intermediate files. This may be necessary because the STL source file and the intermediate file always have the same name, but are not always the same version.

Table 21-1 Special Functions

Function Key	Function
SEQ>INT	Converts a sequential file into an intermediate file. You use this function if, for example, your STL source file was written with another text editor and you want to compile it into a STEP program file, or if the intermediate file no longer exists.
INT>SEQ	Converts an intermediate file into a sequential file. If you have decompiled from a program file and want to edit it using the STL editor, the intermediate file must first be converted into a sequential file. This function is especially important if an old version of the source file exists. This function is also very helpful, when editing the source file with a foreign language STL editor.

Table 21-1 Special Functions, continued

Function Key	Function
SEQDELETE	With the SEQDELETE function, you can delete sequential files, for example if you want to generate a new version during compilation. If the compiled files are edited the sequential files are automatically generated by the editor.
INTDELETE	Intermediate files are deleted using INTDELETE (e.g. old version). They are either recreated using the SEQ>INT function from an updated source file or when you enter an edited source.
COPY	If you want to copy the intermediate file and the STL source file to other drives as backup, use the COPY function. Remember that the files can only be renamed using the F-AUX tools.
TEST RUN	The TEST RUN can be used to test the compiled blocks of a program file, but also blocks from the LAD, CSF, STL package to check whether the commands are permitted for the selected PLC type.
SYM-GEN	The SYM-GEN function creates a symbolic source file from the STL source file, which contains all symbols and absolute parameters used. This symbolic source file can be extended by assignments and comments using the symbolic editor. The symbols and absolute parameters which already exist in the STL source need not be entered again.

21.3.3 Standard Programs

Standard Programs

Because it is possible to write a statement list using only symbols and because the intermediate file is non-language dependent, standard programs can be created.

Debugged blocks and modules can be stored in libraries and then linked with the include command to form individual programs for specific systems. This means that you only need to link your new programs to a special assignment list to obtain system-specific STEP 5 programs for individual control tasks.

21.3.4 Foreign Language STEP 5 Program Versions

Overview

You can also create foreign-language STEP 5 program versions with the batch compiler if you have programmed your program in absolute format or if you have a decompiled intermediate file with absolute parameters. With the English and the French software versions of this package you create English and French STL sources.

If you require these functions, change to the **F-AUX, F-TRANSFER** utility and copy your German STL source file and intermediate file with a new name. You must enter this new name in the presetting of the English/French software version. The **INT>SEQ** special function converts the non-language dependent intermediate file into the sequential source file. This file is then output with English/French STEP 5 commands when you edit it.

Another possibility is to delete the German sequential source file (**SEQDELETE** special function) in the English/French package. A new one is automatically generated when you use the editing function.

Symbols and comments are not output in foreign languages.

21.4 Editing and Adding to STEP 5 Blocks

Creating Blocks with the STL Editor

The name of the STL source file (and possibly also the drive for the intermediate file) is entered in the presettings. After calling the STL editor the selected statement list and the editing menu are displayed on the screen. You can now edit the statement list or add to it using the editing functions. Always save your modified source file with enter, so that the intermediate file is updated and your “new” program file is not created with an “old” program.

If you exit the STL editor with the cancel key and confirm the cancel, the additions or modifications are not entered in the STL source file.

Editing Blocks Using the LAD, CSF, STL Package

Program file blocks must be decompiled before they can be edited in the STL editor. Once they exist as an STL source file, they can be edited in the STL editor as described above.

21.5 Test Run

Program File Testing

The test run takes place after compilation. During the test run the program file blocks are tested (the parameter transfer of function blocks and the existence of called blocks are checked). You select a test run for a block, a group of blocks or all blocks of a program file. If a language category ID has been entered in the presettings, the test run also checks whether the statements are permitted for the PLC type. Illegal statements are logged in the error list.

Testing Special Blocks

Standard function blocks, Graph 5 and assembler blocks cannot be created and decompiled with the STL editor/batch compiler but they can be tested using the test run. The existence and transfer of parameters as well as the validity of the STL statements for the selected PLC type are also checked.

21.6 Error Messages

Error messages can occur in the following phases:

- Compilation of the STL source file into the intermediate file.
- Compilation of the intermediate file into the program file.
- Decompilation of the program file into the intermediate file.
- Decompilation of the intermediate file into the STL source file.
- Program file test (test run).

The error messages are stored by the programming device in an error list in an ??????AF.SEQ error file.

The error file contains only the error list of the last phase of compilation. The error file can be output on the screen or printer if the last phase was completed without errors.

21.7 Entering STEP 5 Statements with other Editors

STL Source File as the Interface

The STL source file can also be created with other editors. These editors must, however, be able to process “real” tabs (09H hex). If not, the initial columns of the subfields must be indicated in the first line of the STL source file using the #TAB control character.

The first six characters of the file name can be selected to suit your purposes. The name must consist of six characters. A0.SEQ is always the last two characters of the name and the extension. This file can only be processed with the tools of the STL editor/batch compiler package without problems if its format matches the format of the sequential source file described below. The STL editor/batch compiler then supports you with the SEQ>INT special function and compiles the file into the program file. Alternatively, you can select direct compilation with the SEQ>MC5 function.

Sequential Source File Format

One data record is entered per statement line. A data block begins with the tab character (09H) and consists of four data fields that are also separated by tabs. The end of a data record is marked with “carriage return, CR” (=0DH) and “line feed, LF” (=0AH). This is automatically added by the editor at the end of a line after you press the Return key. The maximum number of characters for the following fields are as follows:

TAB	TAB	TAB TAB	CR, LF
Address	**Statement**	**Operand symbol**	**Statement comment**
4 chars	13 chars	24 chars	32 chars

This means, that the data block of a blank line consists of four tab characters followed by the “CR” and “LF” characters.

The data record for comment lines starts with the tab character (=09H), directly followed by the control characters ***** and **;** for segment and additional comments. A comment of up to 79 characters can follow and the line is completed with the “CR” (=0DH) and “LF” characters (=0AH).

Lower and upper case letters are allowed in the data blocks. Lower case letters are automatically converted into upper case letters by the editor when they are read in. Accented vowels (umlauts) cannot be used (e.g. ö,é etc.).

#TAB Control Characters for Processing of External Files

The #TAB control character allows files without real tab characters (for example files created in many text programs such as 1st Wordplus) to be compiled. However, the STL editor cannot edit these files and a “wrong file format” error message is output.

#TAB must be placed directly at the beginning of the source file. Only blanks are allowed before it. It must be followed by 4 numbers, separated by commas, which determine the initial columns of the subfields. No further entries are allowed in the first line!

Example:

If 1 blank each is required between the subfields as a separation, the first line of the STL source file is as follows:

#TAB 1,6,21,46 RETURN (CR LF)

The numbers for the columns always relate to the beginning of the line. The difference between the consecutive entries must be at least as high as the corresponding lengths of the subfields.

21.8 Working on the Programming Device

Loading the STL Editor/Batch Compiler Package

Follow the steps outlined below:

1. Select the **Change** menu,
2. Start **AWL Batch**.

The following dialog appears:

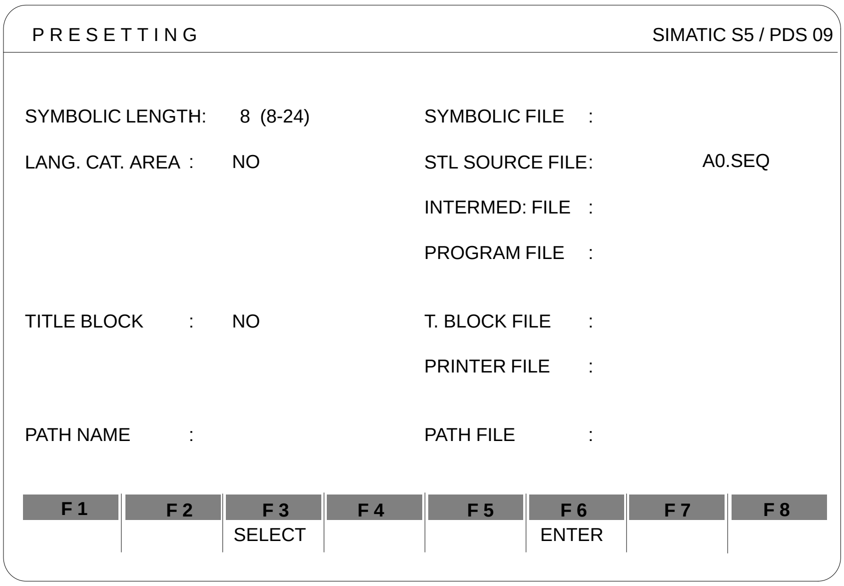


Figure 21-1 Presettings Dialog

Presetting

The cursor is flashing in the line of the STL SOURCE FILE (A0.SEQ). Your statement list is saved in this file. It is marked as a sequential file, i.e. as an ASCII file and it is the source file for the compilation.

Enter the name of your file. In your example the file is named “test”.

1. Enter character sequence *Test.*
2. Press the **Return** key.

A hard disk is entered as the drive by programming device and the name is padded out with @@. The INTERMEDIATE FILE (A1.SEQ), the SYMBOL FILE (Z0.INI) and the PROGRAM FILE (ST.S5D) have the same name to indicate that they belong together.

If no symbols file, in other words no assignment list with this name exists, then this is indicated three times: (GESP) is displayed after the file name, the programming device displays the message “file C:TEST@@Z0.INI: file not available”, and in the SYMBOLIC LENGTH field the default 8 is replaced by 0.

Overwriting Names

If the source file is to be linked to a symbol file with another name during the compilation and/or is to be compiled into a program file with another name, then the individual names can now be overwritten. After overwriting the name press the Return key. These files are also active in other STEP 5 packages and are entered in their presettings when they are loaded. The names of the files for title block, printer and paths are automatically adapted to the program file.

- Position the cursor in the required line.
- Press the **arrow key right**,
- Type in the name.

The PG checks whether the entered files are available. If you want to use other files, enter their names. If the printer and path file are not available, their names are deleted when the cursor is moved up or down.

The PATH NAME and TITLE BLOCK (footer) lines are handled in the same way as in the LAD, CSF, STL package: The name is entered for the path name and the width is selected for the title block.

Language Category

In the LANG. CAT. AREA field you should always use the **Help** key (position the cursor on one letter of the world NO): press the **Help** key to display the programmable controllers (PLCs) and central processing units (CPUs) are for which the batch compiler can compile and test.

If required, enter the device on which your program will run from the list displayed as the language category. The batch compiler then checks whether your statement list corresponds to the PLC language category when it is compiled into the program file.

In the SYMBOLIC LENGTH field, the symbol length of the selected symbols file is displayed. You cannot modify this field.

To save this PRESETTING, press

- ENTER (F6)

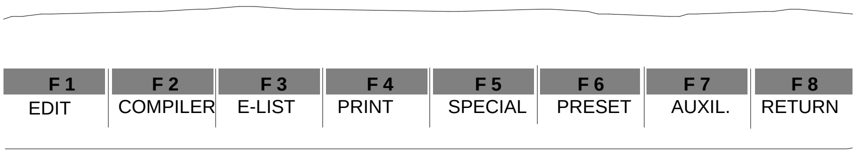
or

- the **Insert** key**.**

The FUNCTION SELECTION is then displayed.

Function Selection

The FUNCTION SELECTION displays the function keys for the following editing and processing functions. The following sections explain how to use them.



21.8.1 Editing

Starting the Edit Mode

Press **EDIT** (F1).

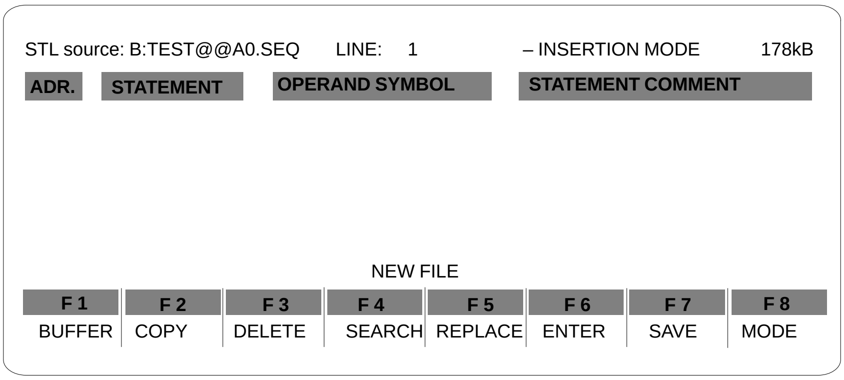


Figure 21-2 Editing Dialog

In this dialog, you can either enter a new statement list or display an existing statement list for editing (corrections, modifications).

Title Bar

This contains the following fields:

- The name of your preset STL source file and the drive.
- The line where the cursor is currently positioned
- The insert or overwrite editing mode and
- The buffer sizes of the memory. This is of interest for processing speed.

Editing Field

The editing field is divided into four columns with fixed widths. The width and the intended contents of the columns are shown briefly below:

ADR 4 characters	STATEMENT 13 characters	OPERAND SYMBOL 24 characters (maximum)	STATEMENT COMMENT 32 characters
Addresses, jump labels	Operations, absolute operands, constants	Symbols Values of the constants	Comments

Message Line

All messages are displayed in the line above the function key bar, for example “new file”, if a new statement list is being created.

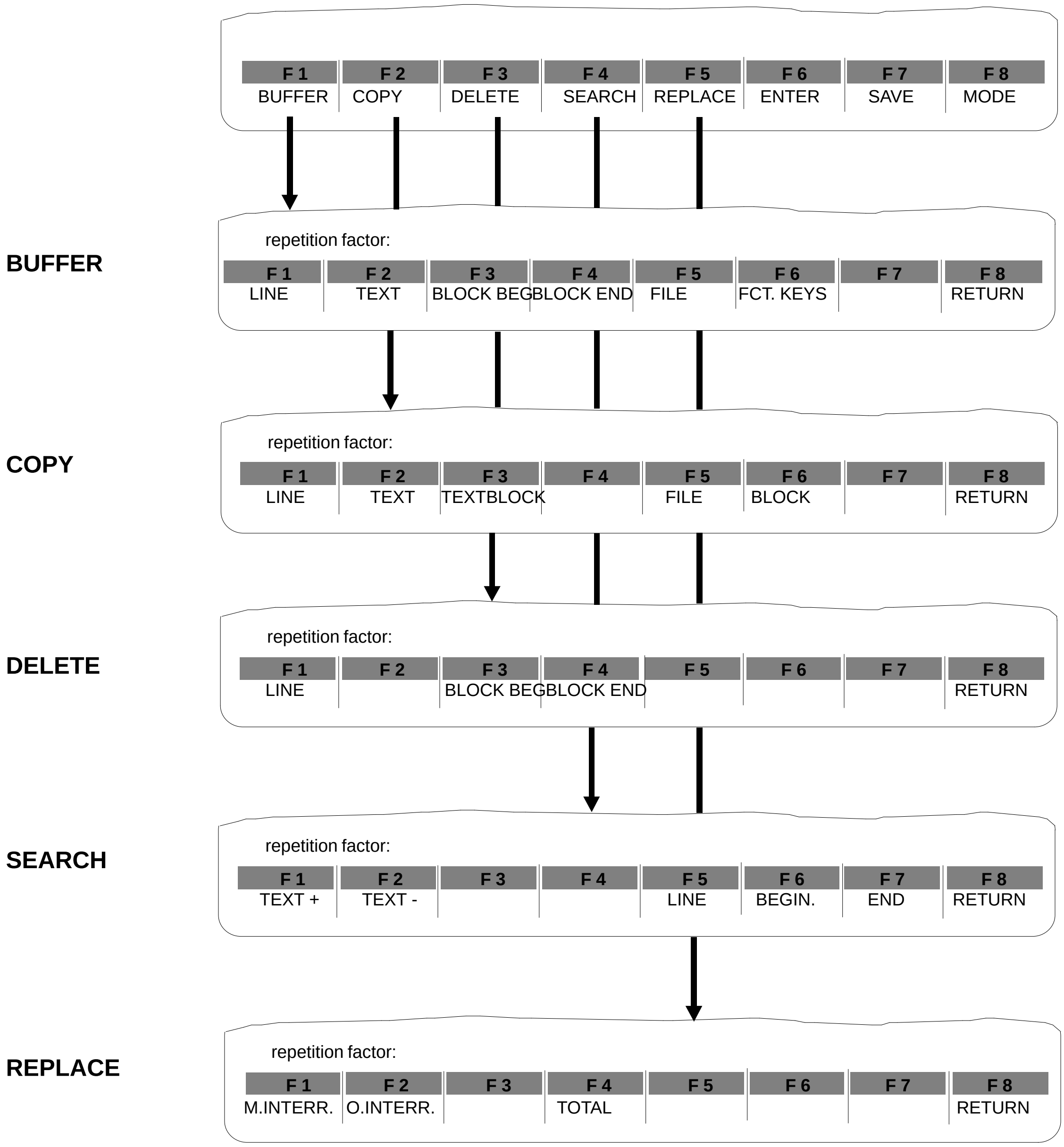
Function Keys

These editing functions are identical to those of the SYMBOL EDITOR. They are used to create and process a statement list.

F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8
EDIT	COMPILER	E-LIST	PRINT	SPECIAL	PRESET	AUXIL.	RETURN

Function Keys for EDIT

The following diagram is an overview of the tools available for the individual editing functions. If you press one of the function keys available in the Edit menu, the function key bar displays the key assignment indicated by the arrow.



Special Keys In addition to these functions the special keys are available as further “tools” for editing your file.

Cursor Keys All the cursor keys can be used to move the cursor.

21.9 The Control Characters of the STL Editor/Batch Compiler

Overview

For certain entries the STL EDITOR requires several control characters so that the statement list can be compiled into a STEP 5 program file. For example segment titles and comments, actual operands and block identifiers must be identified.

These control characters are listed in the table below. This shows the order necessary for problem-free compilation into the intermediate and program files. Among other things, the table also shows the conventions (_represents a blank) and the position of the control characters within the statement list.

Table 21-2 Control Character in the Statement Column

STATEMENT Column Control Characters	Identifier for	Conventions with Examples	Position in the Statement List	Explanations
#TAB	Source file without genuine tabs	#TAB 1,6,21,46	Always the first line of a file	Allows compilation of files created with a different editor, for example 1st Wordplus. Applies only to the compiler, not for the STL Editor.
#TY	PLC type	with blanks #TY_S5-155U #TY_CPU928	Always the first statement in a file	Any comments are only in the STL source file, they are not compiled and are lost if you decompile.
#PBn #OBn #FBn, #FXn #DBn, #DXn (#SBn, not GRAPH 5 block)	Program block start Organization block start Function block start Data block start Sequence block start	without blanks #PB11 #OB1 #FB25, #FX12 #DB5, #DX33 #SB3	Start of a block; after a BE (block end see operations below)	Range of values: n= 0 - 255, depending on the PLC type. If you want to enter further statements after a block end, they must be preceded by a block start otherwise the statements will be lost when the source file is compiled. DB0, DB1, DB2 are not permitted.
#BI	Library number	with blanks #BI_12345 maximum 65535	After the block start or after the block name (see below #N)	For your own library numbers; you cannot and do not need to enter the numbers of standard function blocks. Any comments are located only in the STL source file, they are not compiled and are lost if you decompile.
#N	Name of a function block	with blanks #N_GARAGE max. 6 chars.	Before or after the library number, but at the start of the block	

Table 21-2 Control Character in the Statement Column, continued

STATEMENT Column Control Characters	Identifier for	Conventions with Examples	Position in the Statement List	Explanations
#UB	Segment title	The control character is located in the STATEMENT column, the title text is in the STATEMENT COMMENT column.	Only at the start of a segment	These comment texts are included in the program file. For more information about comments in STEP 5 programs, refer to the STEP 5 description in volume 2 of the manual for your programming device.
( )	Formal parameter type	The formal parameter type must be in parentheses (D) (I)	Directly below the block name	
,	Actual operand for assigning parameters to a function block.	First character in the column; followed immediately by the parameter ,I1.0	Within a block	
#|	Include file	with blank, drive and the first six characters of the file name #|_A:PRACTI	Only at block boundaries: before the first block or between BE and #PBn	This control character allows other files to be linked. These files must be available as intermediate files, i.e. either completed with the enter key in the STL editor or decompiled from a program. Make sure that the same block names do not occur in the files to be included. When the program file is compiled, the last block with the same name overwrites the previous block with the same name. During compilation, the include file is linked to the preset symbols file. This must also be able to supply the include file with assignments.
#	Symbolic block name			

Table 21-3 Control Characters in the ADR Column

ADR Column Control Characters	Identifier for	Conventions with Examples	Position in the Statement List	Explanations
*****	Segment comment	The control character is located only at the start of a segment; any segment title must immediately precede the comment.		
;	Additional comment	The control character is located in the ADR. column. The entire width of the screen is available for the text regardless of the columns.	At any point in the block	These additional comments only exist in the STL source file. They are ignored by the compiler. If you decompile to the same source file, the comments are lost.

21.9.1 STEP 5 Operations in the STL Editor/Batch Compiler and Writing Conventions

All STEP 5 operations are possible in the STL editor/batch compiler. Only the language category of the programmable controller or the CPU creates restrictions. Check the operation list of your device when programming.

The following table, which corresponds to the screen columns, lists the writing conventions for absolute and symbolic programming.

Table 21-4 STEP 5 Operations

	ADDRESS	STATEMENT	OPERAND SYMBOL	STATEMENT COMMENT
Operation with absolute operand		Operation and absolute operand A_I1.2 format-free entry		“open outside” button
With symbolic operand		OperationU	Symbol B–OPN O without hyphen	
Operation with data		Operation and data format L_KT format-free entry	Value of the data 005.2	
Formal operands	Name TIME BO-O MODN max. 4 characters	Type (D) (I) (Q) in parentheses		

Table 21-4 STEP 5 Operations, continued

	ADDRESS	STATEMENT	OPERAND SYMBOL	STATEMENT COMMENT
Data	Address11	Data format KH KF KS or S KG KT KC KY or A KM	Value, 1 data word per line 6248 + 13512‘display‘ Only single quotes, up to 11 data words per line –1169368–38 max. 1 data double word per line 123.1 735 125,018 00011100 11101111	
Operation with formal operand	Operation and formal operand A_=BO-O =_=MODN format-free entry, the formal operand must be immediately preceded by an equality sign.			
Symbolic	control character with operand ,I1.2 ,DW1 without blanks	Symbol MODN		
Data	Control character ,			
Data	control character with data type ,KT	Value 005.2		
Jump labels	Label ON M003			
Relative addresses, data word addresses	17			
Block end			BE	

You can move from column to column using the **Shift + Arrow** keys. If you press the Return key, the cursor always moves to the 1st character of the STATEMENT column.

Symbols

If you program with symbols, remember that in contrast to the CSF, LAD, STL package no hyphen must be placed before the symbol. A block start can only be entered as a symbol if an assignment of the block type and number to a symbol exists. If this does not exist, the block start must be programmed in absolute format, for example #PB3, because the batch compiler requires the exact block type and its number when it creates the intermediate file.

The symbols used in the STL editor must be identical to those in the symbol file. This also applies to blanks: _EMEOFF is not identical to EMEOFF

Further differences to the CSF, LAD, STL package are

- control characters,
- blanks in operations must be entered by the user,
- data constants and values are in different columns.

21.10 Entering Program BLocks

Programming Example

This illustrates how the STL editor/batch compiler works and the functions of this package. The program controls a garage door. It opens or closes from the outside with a key and pushbuttons and from the inside only “open” and “close” pushbuttons are required. The door is closed after a delay of 5 seconds.

STL source: B:TEST@ @A0.SEQ			
ADR.	STATEMENT	OPERAND SYMBOL	STATEMENT COMMENT
	#PB1		OPEN FROM OUTSIDE AND INSIDE
	#UB		
*THE “OPEN OUTSIDE” BUTTON AND THE KEYSWITCH OR THE “OPEN INSIDE” BUTTON			
*START MOTOR UP. THE MOTOR OPERATES UNTIL THE TOP LIMIT SWITCH			
*IS REACHED OR THE EMERGENCY STOP BUTTON IS PRESSED			
	A(		
	A	I 1.2	OPEN OUTSIDE BUTTON
	A	I 1.4	KEYSWITCH
	O	I 1.5	OPEN INSIDE BUTTON
	)		
	AN	I 1.0	LIMIT SWITCH TOP
	S	Q 1.0	MOTOR UP
	***		
	#UB		OPEN FROM OUTSIDE AND INSIDE
*RESET MOTOR UP OUTPUT.			
	O	I 1.0	LIMIT SWITCH TOP
	O	I 1.7	EMERGENCY STOP BUTTON
	R	Q 1.0	MOTOR UP
	***		
	#UB		CLOSE FROM OUTSIDE AND INSIDE
*THE “CLOSE OUTSIDE” BUTTON AND THE KEYSWITCH OR THE “CLOSE INSIDE” BUTTON			
*START MOTOR DOWN WITH A START DELAY OF 5 SECONDS.			
*MOTOR DOWN RUNS UNTIL THE LIMIT SWITCH BOTTOM IS REACHED OR			

```
*THE EMERGENCY STOP BUTTON IS PRESSED.
            A(
            A                               B–CL O
            A                               KEYSW
            O                               B–CL I
            )
            AN                              LIM-BOT
            L           KT                  005.2
            SS                              ON-DEL
            O                               LIM-BOT
            O                               STOP
            R                               ON-DEL
            L                               ON-DEL
            T           FW
            100
            LC                              ON-DEL
            T           FW                  ON-DEL
            102                             MOT-DN
            A
            =
            BE
```

Ready to Start?

The STL editor/batch compiler has been loaded, the presetting has been completed and the editing function has been called.

- Set the **MODE** (F8)

This function can select between two editing modes: insert or overwrite. In the title bar, the PG displays which mode is selected.

- Press **MODE** (F8), until the required mode is activated.

Block Start

Follow the steps below (the character sequences you enter are written in *italics*, the function to be used in bold letters.):

- Enter *#PB1* as the block start.
- Press the **Return** key twice; by inserting this blank line the program has a clearer structure while you are writing it.
- *#UB* for the title of the first segment,
- Press the **shift + arrow right** twice to move to the STATEMENT COMMENT column,
- *Open from outside or inside*
- Press the **Return** key**,**
- Press **shift + arrow left** once to move to the ADDR column,
- Type in ***** as the control character for the segment comment.

Now you can enter the first text of the example. The whole screen width is available for this entry. Complete each line with Return. To begin a new text line you begin as described above with the shift + arrow left key and *****, because the cursor only jumps automatically into the statement column.

If you write in the insert mode, keep a check on the end of the line. The end can otherwise extend beyond the end of the line where it is lost.

The cursor and special keys are available for editing your text. The ***** control character, however, cannot be removed by “delete character” but only by using the **DELETE** and **LINE** functions.

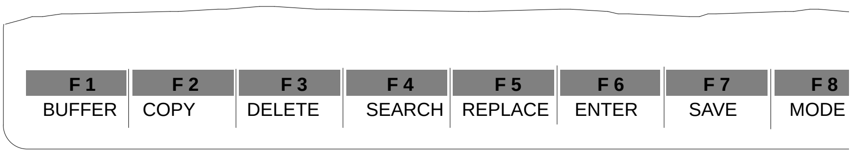
SAVE (F7)

With this function you can save your STL source file without leaving the editor. This means, you can save your work without compiling, for example if you want a break.

This save function differs from the CSF, LAD, STL package, where you always exit the editor after saving.

21.10.1 Using the EDIT Function Keys

Overview



In the explanations of the editing functions, you will see how to use the **BUFFER, COPY** and **DELETE** functions to save parts of your program for further processing in files and how you can assign character sequences to function keys.

The repetition factor is a useful tool: after calling a function, you enter a number at the keyboard of your programming device. The next selected function is executed with this factor, for example, copy a line 7 times.

The function that has been activated is displayed in the title bar above the STATEMENT COMMENT. A function must always be exited with **RETURN** (F8), before you can continue working in the text.

An action is canceled within a function by using the cancel key and/or the F8 **RETURN** key.

Caution

By pressing the cancel key you might lose data! For example, if you correct a file and then press the cancel key all modifications will be lost.

The BUFFER Function

With this function parts of text can be buffered (written to a clipboard). Character sequences (40 characters max.), single lines and character blocks (500 lines max.) are written to separate buffers and can be copied at any location. The text stored temporarily in the buffer can also be saved in a sequential file.

The buffer function is also used to assign character strings to function keys (40 characters max.).

repetition factor:

F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8
LINE	TEXT	BLOCK BEG	BLOCK END	FILE	FCT. KEYS		RETURN

The COPY Function

Texts in the buffer or in sequential files (buffer file, source file) are inserted at the cursor position with this function (paste from clipboard). You can also copy in a complete STL source file at the cursor position.

repetition factor:

F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8
LINE	TEXT	BLOCK		FILE	BLOCK		RETURN

The DELETE Function

With the delete menu you can delete single lines and marked blocks of text. The deleted text is written to the buffer (cut to clipboard) and can be retrieved. However, previously buffered text is lost.

repetition factor:

F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8
LINE		BLOCK BEG	BLOCK END				RETURN

The SEARCH Function

With the SEARCH function you can jump to the beginning and end of your file and to specific lines. Line 0 is not allowed. You can also search for any character sequence up to 20 characters long (words and numbers) in a field. With TEXT + you search forwards from the cursor position for these character sequences and with TEXT – backwards. Using **SEARCH** you can move easily and quickly within your file.

repetition factor:

F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8
TEXT +	TEXT -			LINE	BEGIN.	END	RETURN

Remember that the text to be searched for must be identical to the entered character sequence (case sensitive). This means, you must enter the texts in the “ADDR” and “statement” fields in upper case letters!

The REPLACE Function

Any character sequence up to 20 characters (words or numbers) in the ADDR, OPERAND SYMBOL and STATEMENT COMMENT columns can be replaced by another sequence. You have the option of replacing singly with or without confirmation and replacing all occurrences of the string. Single replacement is only possible from the cursor position downwards. Make sure that you move the cursor up a line before using this function.

Upper/Lower Case

Remember that the text to be searched for must be identical to the entered character sequence (case sensitive). This means, you must enter the texts in the “ADDR” and “statement” fields in upper case letters.

repetition factor:

F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8
M.INTERR.	O.INTERR.		TOTAL				RETURN

Using **REPLACE** you can quickly correct your file. For example, when you want to replace a statement or a symbol in the entire file. Remember that the character sequence you specify must be exactly the same as the sequence in the program including blanks.

Replacing a Character Sequence

If you only want to replace a character string once you can check the replacement by selecting **M. INTERR**; if you select **O. INTERR** the string is replaced without asking for confirmation. The cursor must be positioned above the character string to be replaced because single replacements are made only from the cursor position downwards.

If the text is cannot be found below the cursor position, “not found” is displayed in the message line. Move the cursor to the beginning of the file and repeat the process with **REPLACE** (F5), **M.INTERR** (F1) and the **ENTER** key twice and **YES** (F1). **O.INTERR** starts the same function but without asking you for confirmation.

Repetition Factor

You can combine the single exchange with the repetition factor. The character strings are also found in the comments. They are counted but not replaced, not even if you call **M.INTERR, YES**. Using the **cancel** key you can terminate the replace function before it is finished. The character sequences which have already been replaced remain as they are.

- Exit the **REPLACE** function with **RETURN** (F8) and discard all changes using the
- **cancel** key**,** acknowledge with the
- **ENTER** key. You now return to FUNCTION SELECTION.

Display your original file again:

- **EDIT** (F1)

The SAVE and ENTER Functions

You are already familiar with the **SAVE** function. It can be used for saving your work without compiling and without having to open the file again.

With the **ENTER** function (F6) or the **Insert** key the file is stored, the intermediate file is automatically compiled and the editing session is terminated. During the compilation the created statement list is checked. If more than one error occurs, an error list is created. If only one error occurs, its position is shown on the screen and you can correct it based on the message in the footer.

For more information about the error list, refer to Section 21.15.

Save the file with **ENTER** (F6): it is compiled and editing is completed. The FUNCTION SELECTION dialog is displayed.

21.11 Entering Function Blocks

Example

The C:FBTESTA0.SEQ file, printed out on the next page, can be used as a practice example. Again it is the control of a garage door, but this time programmed as a function block, to show you the differences when editing this type of block.

Here the program call is programmed symbolically. This means that you require the following assignment list in the TEST@@Z0.INI symbol file so that the compilation will work.

SEQ. FILE: TEST@@Z0.INI

I1.0	LIM-TOP	LIMIT SWITCH TOP
I1.1	LIM-BOT	LIMIT SWITCH BOTTOM
I1.2	B–OPN O	OPEN OUTSIDE BUTTON
I1.3	B–CL O	CLOSE OUTSIDE BUTTON
I1.4	KEYSW	KEYSWITCH OUTSIDE
I1.5	B–OPN I	OPEN INSIDE BUTTON
I1.6	B–CL I	CLOSE INSIDE BUTTON
I1.7	STOP	EMERGENCY STOP BUTTON
Q1.0	MOT–UP	MOTOR UP
Q1.1	MOT–DN	MOTOR DOWN
T1	ON–DEL	ON DELAY, 5 SECONDS
FB1	GARAGE	FB FOR CONTROLLING A GARAGE DOOR

STL-source: B: FBTESTA0.SEQ

ADR.	STATEMENT	OPERAND SYMBOL	STATEMENT COMMENT
	#	GARAGE	FB1 FOR A GARAGE DOOR
	#N GARAGE		
LIMT	(I)		LIMIT SWITCH TOP
LIMB	(I)		LIMIT SWITCH BOTTOM
BO-I	(I)		OPEN INSIDE BUTTON
BO-O	(I)		OPEN OUTSIDE BUTTON
BC-I	(I)		CLOSE INSIDE BUTTON
BC-O	(I)		CLOSE OUTSIDE BUTTON
KEYS	(I)		KEYSWITCH
STOP	(I)		EMERGENCY STOP BUTTON
MOUP	(Q)		MOTOR UP
MODN	(Q)		MOTOR DOWN

```
                    #UB                                OPEN FROM OUTSIDE OR
                    AN   =STOP                         INSIDE
                    A(
                    A     =BO-O
                    A    =KEYS
                    O     =BO-I
                    )
                    AN   =ENDO
                    S     =MOUP
                    ***
                    #UB
                                                       CLOSE FROM OUTSIDE OR
                                                       INSIDE
*RESET THE MOTOR UP OUTPUT
                    O    =ENDO
                    O    =STOP
                    RB   =MOUP
                    ***
                    #UB                                CLOSE FROM OUTSIDE OR
                                                       INSIDE
*HERE; THE DOOR CLOSES IMMEDIATELY.
                    A(
                    A     =BC-I
                    A    =KEYS
                    O     =BC-I
                    )
                    AN   =ENDU
                    S     =MODN
                    ***
                    #UB                                CLOSE FROM OUTSIDE OR
                                                       INSIDE
*RESET THE MOTOR DOWN OUTPUT
                    O     =ENDU
                    O     =STOP
                    RB    =MODN
                    BE
```

Ready to Start?

The STL editor/batch compiler package must be loaded.
Return to the example. Complete the PRESETTING dialog with the file name FBTEST for the STL source file and the intermediate file and TEST@@ for the program file and the symbol file. *ENTER* and call the editing function.

If you have not left the STL editor/batch compiler package, the FUNCTION SELECTION dialog of the package is shown.

Return to the PRESETTING dialog and change the name of the STL source file to FBTEST.

Parameter Assignment

To assign parameters to the function block, in other words to provide it with actual operands, you write a program block:

STL source: B: FBTESTA0.SEQ

ADR.	STATEMENT	OPERAND SYMBOL	STATEMENT COMMENT
	#PB2		PARAMETERS FOR FB1
	#JU	GARAGE	
	,I 1.0		
	,I 1.1		
	,	B-OPN I	
	,	B-OPN O	
	,	B–CL I	
	,	B–CL O	
	,	KEYSW	
	,	STOP	
	,	MOT-UP	
	,	MOT-DN	
	BE		

You can enter the actual operands in either absolute or symbolic form.

- Remember that a comma must be placed before every every actual operand,
- and that the order of the parameters must match that of the identifier list of the formal operands in the function block.

21.12 Entering Data Blocks

Overview

The following data block will be basis for this section. If you require information on data blocks, refer to the Introduction and to the chapter on data blocks in the STEP 5 description of your PG manual, volume 2.

STL source: B: FBTESTA0.SEQ

ADR.	STATEMENT	OPERAND SYMBOL	STATEMENT COMMENT
	#DB 12		
0	KH	FFFF	
1	KM	1111111 11000000NUMBER	
2	KH	0013	
3	KF	–32768	
4	KF	+32767	
5	KG	–2740000+22	
7	KG	–1234000+05	
100	KY	022,033	
111	KY	022,033	
	KY	022,033	
	KY	022,033	
	KH	ADAC	
	KH	4538	
	KF	+32767	
	KF	+32767	
	KF	+32767	
	KF	+32767	
	KF	+32767	
	KY	022,033	
	KY	022,033	

Ready to Start?

The STL editor/batch compiler package has been loaded. If you have just worked through the function block example, you are in the editing function and the FBTEST file is displayed.

Return to this example, complete the PRESETTING dialog with the file names FBTEST for the STL source file and intermediate file and TEST@@ for the program and the symbol file. Enter and start the editing function.

Note

If you enter an address, which does not match the actual address in the DB, the space is padded with KH 0000 during the compilation (in the example the address 9 to 99). By doing this, you create space for the data from the process.

The repetition factor cannot be directly used as in the CSF, LAD, STL package but only in conjunction with the **COPY** function.

21.13 Editing an STL Source File

Overview

If you want to edit an STL source file within the STL editor/batch compiler, it is displayed on the screen with EDIT and you can edit it using the editing functions.

In our example the FBTEST file will be included in the TEST@@ STL source file, using the include command. FBTESTA0.SEQ must therefore exist as an intermediate file. We have already done this (see above).

Ready to Start?

TEST@@ has been entered as the STL source file in the presetting dialog.

- **EDIT** (F1) file TEST@@.

Jump to the end of the file with

- **SEARCH** (F4),
- **END** (F7), and then change back to the edit mode with **RETURN** (F8).

The insert mode is selected.

- Move the cursor before the first block, between BE and #PBn or to the file end after the last block end BE;
- Expand vertically; now there is enough space for the include command.
  #I blank *B:FBTEST*
- Press **ENTER** (F6) to save and compile and your intermediate file is updated.

If you now compile the TEST@@A0.SEQ STL source file into the TEST@@ST.S5D STEP 5 program file, the FBTESTA1.SEQ is also compiled and transferred into the program file. All the blocks edited during this practice session are then available there.

21.14 Compiling with the COMPILER Function

Overview

After you store your STL source file with ENTER, it exists as an intermediate file (INT). To compile it into a STEP 5 program, call the **COMPILER** function**.** You can then compile your statement list into the program file you selected in the presetting dialog. With **INT>MC5** the intermediate file is transformed into the MC5 machine code. With **SEQ>MC5** the STL source file is transformed into the MC5 machine code and the intermediate file is generated automatically.

You can decompile in the same way: an intermediate file is created from an MC5 program file with **MC5>INT** (such an intermediate file is further decompiled to the corresponding STL source file with the SPECIAL functions) or you can create both the intermediate file and the source file at one time with **MC5>SEQ.**

The **SEQ>MC5** function first compiles the intermediate file (SEQ>INT). If any errors occur here, the **INT>MC5** compilation is not started and the function is terminated. The error messages resulting from the compilation of the intermediate file are written in the error list. In the same way, the **MC5>SEQ** function first starts the **MC5>INT** compilation and the **INT>SEQ** compilation is only started if the intermediate file is created without any errors.

Operating Sequence: Compiling into the Program File

In the presetting, the FBTESTA0.SEQ STL source file is present.

- Call the **COMPILER** function with F2,
- press **INT>MC5** (F2) or **SEQ>MC5** (F1).
- Fill in the following command line:

Compilation of blocks:	OPT:	PRI:

Press the **help** key to display all possible inputs for each input field.

- Press the **help** key in the Block field:

In addition to the usual inputs in the STEP 5 basic package you can enter block ranges to be edited, e.g. PB12 –PB21.

- Type B in this field and complete with the **Return** key**.**
- Press the **help** key in the OPT field.

With “2” you can start a compilation test: your intermediate file is compiled and checked for errors. It is not stored in the program file, however. Any errors can be seen in the error list.

- Type 2 and press the **Return** key.
- If your program file already contains blocks with identical names and the OPT field is empty, you are prompted to confirm before overwriting a block; when confirmed, the old block will be overwritten by the new one with an identical name. If you use the option “1”, the blocks are overwritten in the program file without your confirmation.
- Press the **help** key in the PRI field.

The printer output formats are the usual ones for the STEP 5 basic package: standard print, condensed print and super condensed print. The paper size DIN A3 or A4 depends on the connected printer.

In our example this field remains empty.

Compilation of blocks: B	OPT: 2	IMP:

- Press the **Insert** key.

The programming device now compiles the blocks and tests them. The names of the blocks that have been processed and how many errors have occurred is displayed or a message indicating that there were no errors in compilation. The function selection dialog is displayed again.

If no errors occurred during compilation, then repeat the compilation. Leave the option field empty so that the program file is created and enter ***** in the PRI field to obtain a printout.

Compilation of blocks: B	OPT:	IMP: *

- Press the **Insert** key**.**

The blocks of the FBTESTA0.SEQ STL source file are then transferred to the TEST@@ST.S5D program file in machine code and saved. Now you can continue working with FB1 and PB2 in the CSF, LAD, STL package (e.g. test on the PLC).

How to Decompile the Program File

This is basically the same as compilation. You simply use the **MC5>INT** (F4) or **MC5>SEQ** (F5) function. It is important for the decompilation that the required files are set in the presetting. For the command line the same conventions apply as when compiling. With the help key in the OPT field the following transfer sequence is possible:

- start **COMPILER** (2),
- select **MC5>INT** (F4),
- complete in the command line,
- press the **Return** key.
- The intermediate file is now recreated or one with the same name is overwritten (after confirmation). The **MC5>SEQ** function automatically generates the STL source file, which you can edit with the STL editor. With the **MC5>INT** function you must create a sequential source file yourself using the INT>SEQ SPECIAL function (see Section 21.17).

21.15 Error List

The error list is not only a list of the errors which occurred during the compilation, but it is also a complete log of the compilation: it lists the blocks compiled without errors and if compilation was canceled it indicates where this occurred.

To have a real example, create an error in your FB 1 (programmed in Section 21.11): **EDIT** your FB 1 and write, for example, only R as reset commands. When you save with **ENTER**, a message tells you that errors have occurred.

1. Now call the **E-LIST** (F3).
2. Complete the PRI input field in the command line so that you can easily correct the STL source based on the printout. If you press the help key, the parameters for the PRI field are displayed.
3. Press the **Insert** key: Each incorrect statement is shown with a block identifier, a line number and an explanation. The blocks that were compiled correctly are also listed.

```
FILE B:FBTESTAF.SEQ
COMPILATION STL SOURCE B:FBTESTA0.SEQ => INTERMED. FILE B:FBTESTA1.SEQ
                                    R =MOUP
*** ERROR IN LINE                   28: OPERAND ILLEGAL  ***
                                    R  =MODN
*** ERROR IN LINE                   45: OPERAND ILLEGAL ***
*** FB1 COMPILED,                   2 ERRORS FOUND ***
*** PB1 COMPILED, BLOCK ERROR FREE ***
*** COMPILATION COMPLETED; 2 ERRORS, NO WARNINGS ***
```

Displaying the Error List

The error list is displayed on the screen if you leave the PRI field in the command line empty. If long error lists are displayed on the screen, the pages are 20 lines long and you can either cancel the display by pressing the cancel key or view the next screen page using the insert key.

21.16 Printing

Overview

With this function you only print out the preset STL source file. In the command line, you simply need to select the layout of your printout. (You will find detailed information on the printed layout in the STEP 5 description, volume 2, in the chapter dealing with input/output of STL blocks.) Compiled files can only be printed out with the command lines of the COMPILER function.

Ready to Start?

The printer is connected and ready for operation; if you have an external printer, you must set parameters in the PRINTER UTILITY PROGram. The PT 88 printer is the default. The name of the file to be printed is entered in the presetting of the STL editor/batch compiler, for example, FBTEST. The function selection dialog is displayed on the screen.

How to Use the Print Function

Follow the steps outlined below:

- Press **PRINT** (F4)
- Fill in the PRI field; the default is standard output in standard print.
- Press the **ENTER** key**.**

The FBTESTAQ.SEQ STL source file is printed. The function selection dialog is displayed again.

As in the CSF, LAD, STL package, you can also redirect your printout to a file. You define this file in the PRINTER UTILITY PROGram and then enter the name of the printer file in the PRESETTING. For your current printing job, select a layout as usual in the command line. This layout is then also transferred to the print file.

21.17 SPECIAL Functions for Editing Intermediate Files and Source Files

Overview

The **special** functions are used to edit and modify sequential files and intermediate files. You also have the option of a test run for the compiled program file. All these activities relate to the files selected in the presetting. Make sure that the settings are correct before using the functions. In our example the presetting remains unchanged.

Any function can be stopped with the cancel key.

F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8
SEQ>INT	INT>SEQ	SEQDELETE	INTDELETE	COPY	TEST RUN	SYM-GEN	RETURN

COPY

Use the COPY function to create back-up copies. Initially, you can copy the intermediate file and then copy the STL source file to another drive. The programming device displays a message, for example "hardware error" if a floppy disk drive is not closed.

To edit the example files without any risk, first copy them to a floppy disk.

- Enter the STL source file in the PRESETTING,
- Start **SPECIAL** functions (F5),
- Start **COPY** (F5),
- Type in drive: *A*
- Press the **Insert** key. The intermediate file is also saved on a floppy disk. When the question appears "Also copy SEQ.source file?"
- Press the **Insert** key (yes).

The intermediate file and the STL source are copied to the floppy disk. The function selection dialog is displayed again on the programming device.

SEQ>INT

Start the **SEQ>INT** (F1) function, if, for example, you want to compile an STL source file that was created with another text editor. With this function the text file is converted into an intermediate file before it can be compiled into the program file.

- Enter the text file in the PRESETTING,
- Activate **SPECIAL** functions (F5),
- Start the **SEQ>INT** (F1) function. The device displays:" Compilation of the STL source file into the intermediate file?"
- Press the **Insert** key (yes).

An intermediate file is now available for further editing. The function selection dialog is displayed again.

INT>SEQ

Start the **INT>SEQ** (F2) function, if, for example, you decompiled a program file (with COMPILER, MC5>INT) and you want to edit it in the editor. For this purpose you have to convert the intermediate file into a sequential file. Additional comments of the former STL source are lost during this process.

- Specify the required files in the PRESETTING dialog.
- Activate **SPECIAL** functions (F5),
- Start the **INT>SEQ** (F2) function. The device asks: “Compilation of the intermediate file into the STL source file?”
- Press the **Insert** key (yes).

A new STL source file is then created. The function selection dialog is displayed again.

SEQ DEL and INT DEL

SEQ DEL and **INT DEL** delete the preset STL source file and the intermeiate file.

- Specify the required files in the PRESETTING dialog.
- Activate **SPECIAL** functions (F5),
- Start the **SEQ DEL** (F3) function. The device asks: “Delete the STL source file?”
- Press the **Insert** key (yes).
- The function selection dialog is displayed again.
- Activate **SPECIAL** functions (F5),
- Start the **INT DEL** (F4) function. The device asks: “Delete the intermediate file?”
- Press the **Insert** key (yes) or cancel (no).

The function selection dialog is displayed again.

TEST RUN

The **TEST RUN** is additional block check in the preset program file. The test checks whether the standard function blocks are supplied with the correct parameters. Any errors detected are written to the error list.

In the command line of this function you can enter single blocks, block groups, block types or all blocks from a program file; use the help key if you require more information.

- Specify the program file to be tested and, if required, the PLC type in the PRESETTING dialog.
- Activate **SPECIAL** functions (F5),
- Start the **TEST RUN** (F6) function,
- Complete the command line: for example *****,
- Press the **Insert** key**,**
- Select PB1 in the block list.
- Press the **Return** key,
- Enter *FB1, DB12* in the same way,
- Press the **ENTER** key.

Messages are displayed by the programming device during the run. If errors are reported, print out the error list.

SYM-GEN

From an STL source file SYM-GEN generates a symbolic source file, which contains all the absolute parameters and symbols used. You can edit the symbolic source file with the symbols editor to extend assignments and enter comments. The symbols and absolute parameters appear in the symbolic source file as often as they were used in the STL source file. To avoid multiple entries, follow the steps below:

- Create the symbolic source file with SYM-GEN

Example:

Absolute	Symbol	Comment
	EME-OFF	
	EME-OFF	
	EME-OFF	

- Change to the STEP 5 assignment list editor
- Complete the assignment when the symbol or the absolute parameter occurs for the first time

Example:

Absolute	Symbol	Comment
I 1.0	EME-OFF	emergency off
	EME-OFF	
	EME-OFF	

- Compile into the symbols file. Here you can ignore the “Symbol already exists” error messages.
- Decompile the symbols file into the symbolic source file (INT>SEQ)

Example:

Absolute	Symbol	Comment
I 1.0	EME-OFF	emergency off

Now the symbolic source file contains only one assignment with the EME-OFF symbol.

21.18 Error Messages

PLC type illegal: You have specified an invalid PLC type.
Absolute parameter too long: System error! Format or the STL source file incorrect.
Actual parameter not permitted: Specifying actual parameters is only allowed after an FB call.
LIB no. already exists: The control character #BI is specified more than once.
LIB no. invalid: Library number too long, or contains illegal characters (max. 5 numbers).
Block in intermediate file not error-free: Intermediate file (A1.SEQ) is defective (format error). Recreate intermediate file again from the sequential working (A0.SEQ) file with the function **SEQ>INT**.
Block without BE: Command BE (block end identifier) missing
Block too long: Distribute program (max. 8 Kbytes).
Block start missing No # character with absolute and/or symbolic block identifier.
Block type undefined (symbol not found): In purely symbolic programming, the symbolic block identifier is missing.
Block name already exists: The control character #N is specified more than once.
Command for PLC type not permitted: Command for specified PLC type not permitted
Command in block not permitted: Commands from the extended operations are only permitted in FBs.
Command not defined Not a permitted MC5 command. Program file (ST.S5D) defective.
Command not permitted: Not a permitted STEP 5 command.
DB address invalid: DB address too long, or contains illegal characters (max. 5 characters).
Data position wrong: With constants, the value must be specified in the “OPERAND SYMBOL” field.
Doc block too long: Distribute or shorten program documentation (max. 16 Kbytes).
DVS file not opened
Wrong function number: System error!
Wrong nesting depth: Final parenthesis does not match (check nesting levels).

Wrong data record in SEQ-file: Sequential working file, STL source file or intermediate file defective (format error).
Wrong format: Error in format
Error when compiling: System error!
Error when converting: Numeric range was exceeded.
Formal parameter not defined Parameter name and parameter type not defined in FB.
Formal parameter already exists: The parameter name is assigned more than once.
Formal parameter invalid: The parameter name contains invalid characters, or the parameter type is not permitted.
No actual parameter specified: Following the FB call, the required actual parameter is missing in the “STATEMENT” field.
No block name specified: The control character #N and the block name are missing in the function block.
No data specified: After specifying a constant type in the “STATEMENT” field, the constant value is not specified in the “OPERAND SYMBOL” field.
No formal operand specified: The formal parameter declaration in the “ADR” field of the FB is missing for the specified actual parameter after the FB call.
No operand ID specified: Operand ID missing.
No parameter specified: Parameter missing (when programming in absolute format only).
No Symbol specified: Operand symbol missing (when programming in symbolic format only).
No LIB number specified: The library number is missing after #BI.
Comment too long: System error! Format of the STL source file incorrect. statement comment too long (max. 32 characters).
Comment block too long: Distribute or shorten program (max. 16 Kbytes).
Read error Diskette error, file defective.
Label not defined Jump label to specified jump destination (symbolic address) in “ADR” field not entered.
Label invalid: Jump label with illegal character
Label illegal: Jump label at illegal position

Label too long: The jump destination entered in the “STATEMENT” field (symbolic address) is too long (max. 4 characters).
Labels identical: Jump label exists more than once.
More comments than statements
Segment end missing or segment is too long: Segment end character *** or screen refresh command for segment end (BLD 255) missing, or segment is too long (max. 255 lines).
Only permitted with STL blocks: Data block
Only with function blocks: Extended command set not permitted
Only one title per segment: The control character #UB is specified more than once at the start of the segment.
Only after FB call: Actual parameters are permitted only immediately after an FB call.
Operand illegal: No operand permitted
Operand too long: Operand ID too long (max. 2 characters).
Operand ID not defined Operand ID not defined in STEP 5.
Operand ID illegal: Operand ID does not match operator.
Operator not specified: Operator missing in symbolic programming.
Operator invalid: Operator not defined in STEP 5.
Operator too long: (max. 3 characters)
Parameter incorrect: Invalid parameter.
Parameter illegal: No parameter permitted
Parameter too long (max. 4 characters): The formal parameter entered in the “STATEMENT” field is too long (max. 4 characters).
Number of parameters wrong: Declared number of formal parameters in FB is different from the specified number of actual parameters after FB call. (test run)
Parameter range of the PLC type exceeded: This parameter value is not permitted with the specified PLC type
Parameter type wrong: The formal parameter type specified in the “STATEMENT” field is difference from the assigned actual parameter type after FB call. (test run)
SYS commands not permitted: omitted
Control character invalid: The #character is followed by an illegal control character.

Symbol not permitted: Command does not permit operand.
Symbol does not match absolute parameter: Absolute operand and symbolic operand assigned differently in STL source file and symbols file.
Symbol too long: System error! Format of the STL source file incorrect. (max. 24 characters)
Symbols file does not exist: Symbols file missing when programming in symbolic format only.
System commands not permitted: omitted
Character invalid: Invalid character used
Line not permitted: Remember the order (control character) when entering blocks.
Line invalid: Remember the order (control character) when entering blocks.
Line not processed: Block type is undefined.
Too many actual parameters: (max. 40)
Too many formal parameters: (max. 40)
Intermediate file already exists, delete?: An intermediate file with the identical file name already exists.
Intermediate file not error-free: Intermediate file (A1.SEQ) is defective (format error). Compile intermediate file again from the working sequential file (A0.SEQ) with the function **SEQ>INT**.
Intermediate file format invalid: Intermediate file (A1.SEQ) is defective (format error). Compile intermediate file again from the working sequential file (A0.SEQ) with the function **SEQ>INT**.
Intermediate file ID wrong: File was created with tool of a different version. Compile intermediate file again from the working sequential file (A0.SEQ) with the function **SEQ>INT**.
Nested include command not permitted. An intermediate file included with #INCLUDE contains a further #INCLUDE statement.
Symbolic source file exists, overwrite? A symbolic source file with the identical file name already exists (SYM-GEN function).

Parameter Assignment with COM DB1 22

Overview

With the COM DB1 parameter assignment software, you can assign parameters to CPUs of the low to mid range of performance. The time required for successful parameter assignment is minimal.

Up to now, it was only possible to assign parameters to the CPUs in plain text using DB1. To edit DB1 in plain text, you had to use the DB editor of the STEP 5 package.

Chapter Overview

22.1 Range of Functions of COM DB1

Overview

This section covers the following topics:

- The functions provided by COM DB1 and restrictions in the use of the software.
- The CPUs for which you can assign parameters using COM DB1.

Advantages of Assigning Parameters to DB1 with COM DB1

Using this package has the following advantages:

- COM DB1 can interpret and modify every DB1 with parameter assignment data and provide it with comments.
- You no longer need to keep to the rules for DB1 parameter assignment as explained in the PLC manuals since COM DB1 itself takes these rules into account. You can see the CPU-specific parameters on the screen. The arguments and the value ranges of the arguments are available in special list 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 it is transferred to the PLC or to the program file. This excludes the possibility of setting incorrect 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, in other words, a DB1 can be transferred online to the CPU. A DB1 can also be uploaded online from the CPU to the programming device.
- You can display a help text related to the current activity at any point during parameter assignment.

What Does the COM DB1 Software Package Include?

COM DB1 is supplied with STEP 5/ST. It is in the directory `\STEP5\S5_SYS\S5_COM\COM_DB1`.

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

22.1.1 What Functions Does COM DB1 Provide?

The COM DB1 parameter assignment software is a user-friendly tool for assigning parameters to CPUs with a lower to mid range of performance.

The functions provided by COM DB1 are described below. Some functions can only be executed with the CPU online. These are indicated in the text. All other functions can be used both online and offline. You select the online or offline mode in the *Defaults* dialog of COM DB1.

Creating a New DB1

You have just edited a DB1 with COM DB1 and want to discard it. Press the ***F1*** key (New DB1) in the *Overview table* dialog. The DB1 you hav ejust created is deleted and the parameter settings of the default DB1 appear in the Overview table.

You can modify parameters in a DB1 that already exists on the PLC by selecting “Online”, uploading the DB1 from the PLC and overwriting the required parameters.

You can modify parameters in a DB1 that already exists in a STEP 5 program file. You select the STEP 5 program file either in the *Defaults* dialog or in the “*Loading DB1*” dialog. You then load the DB1 from the STEP 5 program file and overwrite the required parameters.

Creating Empty Blocks

When you specify a DB in a parameter block, COM DB1 checks to see if this DB already exists in the PLC (only possible online) or in a STEP 5 program file. If the DB exists but it is not long enough for the parameter assignment, the length is corrected (for example send mailbox DB with SINEC L1).

Comments

You can enter a comment relating to the entire DB1 and to the individual parameter blocks. A comment can consist of up to 80 characters (including spaces).

Transferring a DB1

You can transfer a DB1 to the PLC if you have selected *Online*. If there is already a DB1 on the PLC, you will be asked whether or not you want to overwrite it.

You can transfer a DB1 to a STEP 5 program file. Specify the STEP 5 program file either in the *Defaults* dialog or in the *Transferring DB1* dialog.

Outputting a DB1 to a Printer

You can print DB1 parameters. All parameter assignment dialogs and the *Overview table* 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, (created earlier with the STEP 5 package). You specify the printer file or footer file in the *Defaults* dialog.

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 programming device. You specify the output file in the *Defaults* dialog.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 directly 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.

PLC functions

You can execute PLC functions *online*:

- Compress the PLC memory
- Switch the PLC from STOP to RUN, the DB1 parameters are updated in the CPU
- Switch the PLC from RUN to STOP

Help

COM DB1 also provides a range of Help functions to make parameter assignment easier.

Incorrect parameter assignment is prevented since COM DB1:

- Detects errors as parameters are entered
- Checks 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 prompts you to correct the error (an incorrect DB1 cannot be saved).

22.1.2 Special Features of COM DB1

Please note the following special features and restrictions:

- COM DB1 can only process one DB1 at a time.
- COM DB1 cannot check the interdependencies of parameters between different PLCs (for example 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 set in DB1 can be assigned parameters with COM DB1.
- If a parameter block in the *Overview table* of COM DB1 contains no values, the operating system of your PLC automatically writes the 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 *#* come immediately before the DB1 end-of-text identifier *END*, these characters will be interpreted as comments for the entire DB1.)
- The PLCs listed in Section 22.1.3 can be assigned parameters with COM DB1. The following rules apply to later versions of 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 later version of a PLC, COM DB1 can only set 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.

22.1.3 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 order no. and version (or higher)	
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

22.2 Working with COM DB1

Starting COM DB1

COM DB1 can be started as follows:

1. Start STEP 5.
2. Load COM DB1 with the function **Change ▸ COM DB1**.

The language menu appears on the programming device screen.

22.2.1 Hierarchy of COM DB1 Display Levels

Overview

This section explains how to set parameters with COM DB1 (general operation), how the COM DB1 dialogs are structured on the screen, how to make entries in the COM DB1 dialogs and the rules for making entries. The section also covers the help options provided by the package and error messages that might be displayed.

You work with COM DB1 in dialogs organized into different levels. The following applies to all 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 dialog.
- You can exit every COM DB1 dialog with the ***F8*** *= Return* function key and return to the next higher dialog.

The following diagram illstrates the operating concept when working with COM DB1.

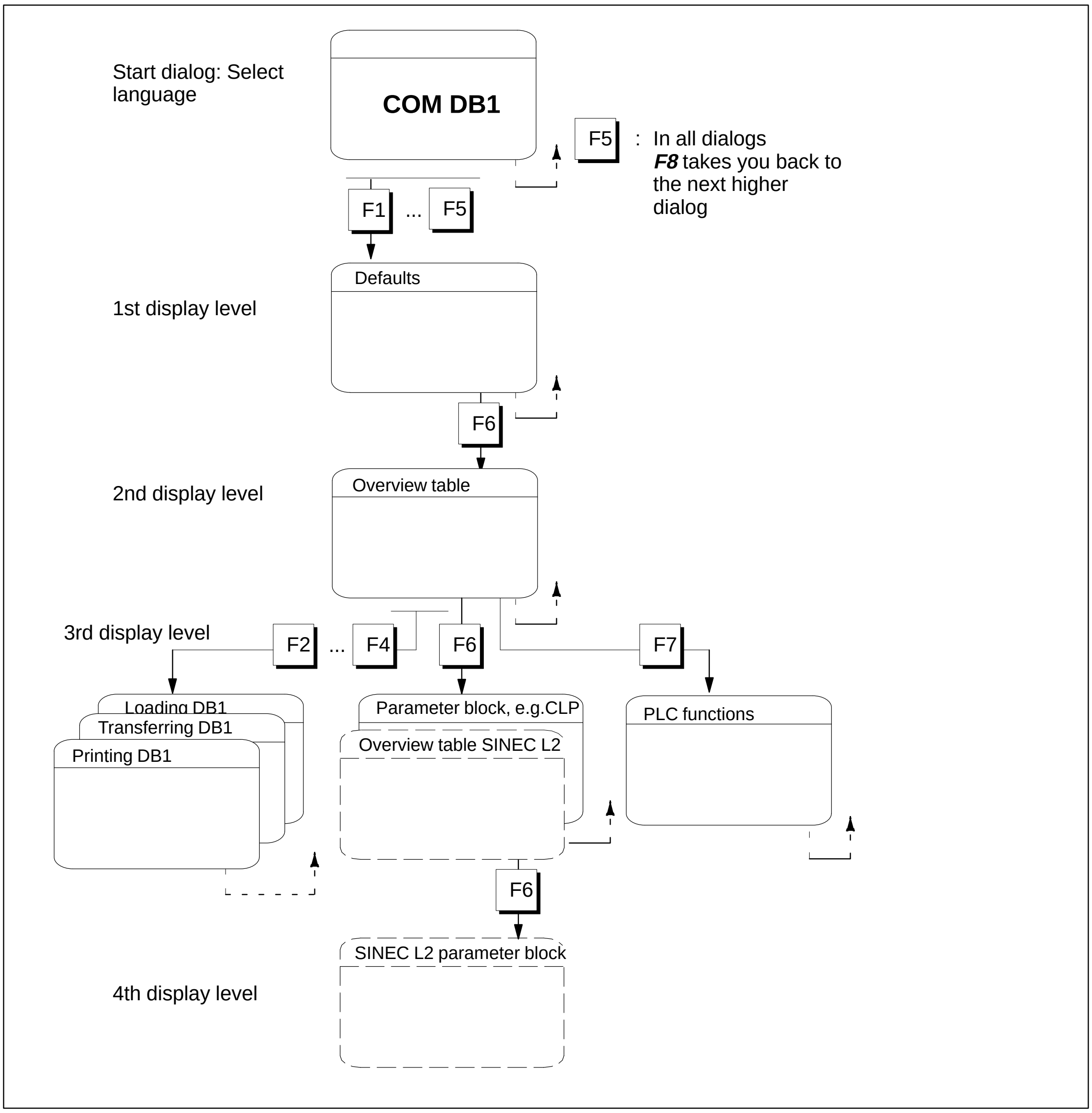


Figure 22-1 Hierarchical Structure of COM DB1

After starting COM DB1, the first COM DB1 dialog appears. This is the *Language* menu. Use the function keys to select COM DB1 in the desired language.

1st Display Level: Defaults

After selecting the language, the Defaults dialog is displayed. Here, you enter the settings required by COM DB1 to execute its functions.

You specify the following in the Defaults dialog:

- How COM DB1 communicates withthe CPU (online, offline)
- Order number of the CPU
- Revision level of the PLC.

Entries in the other input fields of the Defaults dialog depend on the functions you want to execute in the subsequent dialogs. (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).

2nd Display Level: Overview Table

When the defaults have been entered, the *Overview table* dialog is displayed. The Overview table contains all the parameter blocks possible for the CPU type defined in the Defaults dialog. The *Setting* appears beside each individual parameter block (e.g. *Not parameterized*, *Parameterized (default)*, etc.).

You can decide the following in the *Overview table* dialog:

- If you want to load, transfer or print a DB1 that exists in the PLC or in a program file (DB1 utilities)
- 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 change from the Defaults dialog to the *Overview table* dialog, a message tells you whether there is a DB1 in a program file and/or on the PLC. If you load an existing DB1, the Overview table will be updated.

3rd Display Level: DB1 Utilities/Parameter Block.../PLC Functions

If you have selected a DB1 utility function (e.g. *Loading DB1*) or a PLC function at the 2nd display level, the relevant dialog for executing the function appears then at the 3rd display level.

If you have selected a parameter block at the 2nd display level, you branch to the parameter assignment dialog at the 3rd display level. The parameter assignment dialog contains a list of all the parameters belonging to the parameter block. Existing parameter assignment data (e.g. after loading a DB1) appears in the relevant input fields of the parameter assignment dialog. Some input fields without parameters assigned contain default values.

SPECIAL CASE

3rd display level: SINEC L2 Overview table

One screen page 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* dialog, you branch to the *SINEC L2 Overview table* dialog containing the logical subunits.

4th Display Level: SINEC L2 Parameter Block

The fourth display level only exists if the *SINEC L2 Overview table* dialog with the logical subunits appears at the 3rd display level of COM DB. Each subunit has its own parameter assignment dialog. At the 4th display level, *SINEC L2 Parameter Block*, the same entries can be made as at the 3rd display level “Parameter Block...”.

22.3 Layout of the COM DB1 Dialogs

Overview

All COM DB1 functions can be executed by making entries in dialogs. The COM DB1 dialogs all share the same basic layout. They are divided into five areas. The example below of the *Clock Parameters (CLP*) parameter assignment dialog shows the divisions of COM DB1 dialogs.

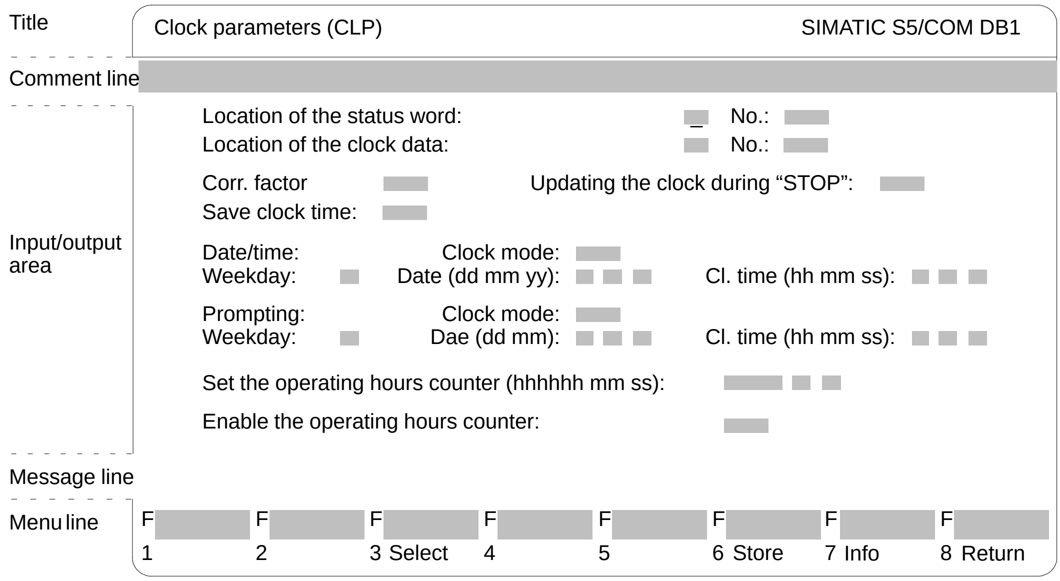


Figure 22-2 Layout of the COM DB1 “Clock Parameters (CLP) Dialog

Title

The titles of all COM DB1 dialogs are one line long and separated from the rest of the dialog area by one line. It indicates the contents of the COM DB1 dialog. The title cannot be changed in any COM DB1 dialog.

Comment Line

Here you can enter a comment relating to the parameter block (in the relevant parameter assignment dialog) or to the entire DB1 (in the *Overview table* dialog). 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 dialogs. This area contains fixed texts and input fields, depending on the display level, in which parameters can be set. Using the keyboard, you can enter the relevant and permitted 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 on the PLC (output area). This is also the area where COM DB1 displays list boxes, help windows and warnings to help you when working with COM DB1.

Message Line

COM DB1 uses the message line to inform you about current processes, operator errors or faults. The first time you change from the Defaults dialog to the *Overview table* dialog, COM DB1 tells you whether a DB1 exists in a program file and/or in the PLC.

Menu Line

The menu line (function keys ***F1*** to ***F8***) at 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.

22.3.1 Possible Entries in COM DB1 Dialogs and Rules to Follow

Overview

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 dialogs 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 at the keyboard.
- 2 Selecting the text from a list 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 settings for the block are free of errors. After the data is stored, COM DB1 switches automatically to the *Overview table* dialog.

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 at the keyboard (e.g. *9*).
3. Complete the entry by pressing the ***Return*** or or ***INSERT*** key. (Press ***ESC*** to discard the text.)

Example of 2**:** Entering the day of the week via a list box

1. Position the cursor on the *Weekday:* input field.
2. Open the list box belonging to the input field by pressing ***F3*** = *Select.*
3. Position the cursor on the relevant text line in the list box.
4. Enter the selected weekday in the input field by pressingthe ***Return*** or ***INSERT*** key. The selected text appears in the input field. (Press ***ESC*** to cancel the entry.)

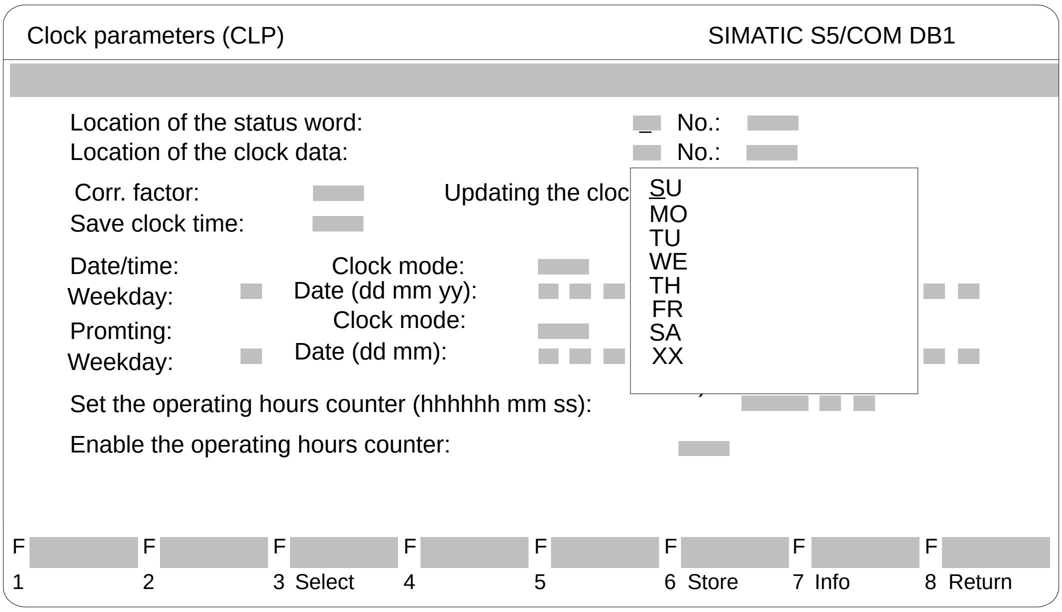


Figure 22-3 COM DB1 *Clock Parameters (CLP)* Dialog: Selecting the Weekday

Entering Comments

With COM DB1, you can enter

1 Comments relating to the entire DB1 in the *Overview Table* dialog and

2 Comments relating to each parameter block in the relevant parameter assignment dialog.

You enter the comment in the comment line provided at the top edge of the COM DB1 dialog. The comment can be up to 80 characters long (including spaces).

Example of 2: Entering a comment for the *Clock Parameters (CLP)* parameter block

1. Press the ***COM*** comment key in the *Clock Parameters (CLP*) parameter assignment form. The cursor then jumps to the comment line.
2. Enter the comment at the keyboard (e.g. *Setting the interrupt interval of maintenance unit 1*).
3. Terminate the entry by pressing the ***Return*** or ***INSERT*** key. (Press ***ESC*** to exit the comment line without changing the original contents.)

Note

A comment relating to 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 Dialogs

below, we have collected a few points to remember and rules for setting parameters for DB1 with COM DB1.

Note

- If you do **not** enter the revision level of the CPU in the Defaults dialog, 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 dialog.
- In the case of CPU 944 with two serial interfaces, you must also specify the order number and the version of the operating system module in the Defaults dialog.
- When loading a DB1 created with STEP 5, comments may be lost if:
  - the comment is longer than 80 characters
  - the comment relating to the entire DB1 is not located immediately before the *END* end-of-text identifier
  - the comment relating to 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 dialog is selected, the default value appears in the input field of the parameter.

22.3.2 COM DB1 Help and Error Handling Concept

Overview

COM DB1 supports you with an extensive help and error handling concept when programming DB1. This section gives you an overview of the following:

- All the help information which COM DB1 provides during parameter assignment
- All error messages which COM DB1 displays when programming 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 dialog 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 dialogs
2. Help screen: Help texts with explanations of the current COM DB1 dialog and function key assignments
3. Info window: Help texts with information on the input fields

Message line 1

COM DB1 informs you about the following in the message line of the COM DB1 dialogs (see Figure *22-4*):

- 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 dialog to the *Overview table* dialog.

Help Dialog 2

If you press the ***HELP*** **key** inside a COM DB1 dialog, a help window appears on the screen with a short explanation of the selected dialog 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 ***Return*** keys.

Press the ***ESC*** **key** to exit the help dialog. The old screen contents are restored.

Example

Help dialog: Explanations of the current COM DB1 *Clock Parameters (CLP)* dialog 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 list 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 relating to the entire parameter block
<ESC> Closes the window

F1	F2	F3	F4	F5	F6	F7	F8
		Select			Store	Info	Return

Figure 22-4 Help Display: Explanations of the Current COM DB1 “Clock Parameters” Dialog and Function Key Assignments

Info Window 3

You can request help information about the input fields of COM DB1 by pressing function key ***F7*** *= Info* (if active) . Depending on the cursor position, all possible and permitted inputs are briefly described in an info window.

In contrast to the help dialogs for explaining function key assignments, each info window only appears as a “popup” so that the input field remains visible.

Only one info window can be opened at a time.

The info window must be closed before making entries 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)* dialog.

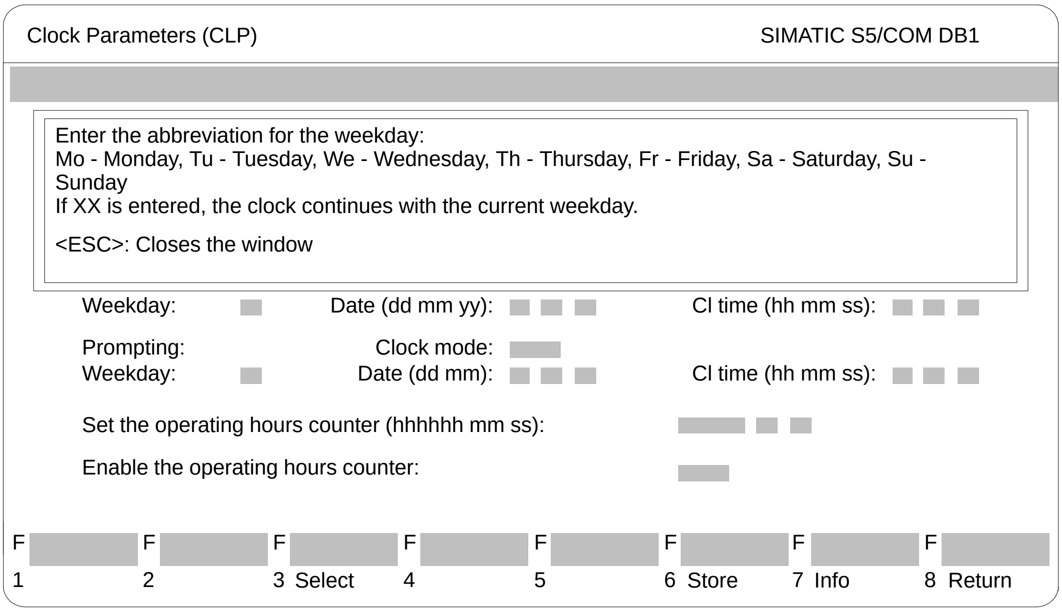


Figure 22-5 Info Window: Information about the *Weekday* Input Field of the COM DB1 *Clock Parameters (CLP)* Dialog

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 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 (input errors)

COM DB1 reacts to these 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?`). Such prompts must be acknowledged with ***ESC*** or answered according to the prompt text with ***ESC*** for *No*, or *Cancel* or ***Return*** for *Yes*.

Errors detected during loading 1

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 displays 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 that was programmed with the DB editor of the STEP 5 package).

Note

If you position the cursor on the incorrect (*!*) parameter in the parameter assignment dialog, the relevant error message will appear in the message line.

Example

Marking incorrect parameters in the *Clock Parameters (CLP)* block after loading DB1. DB1 has been created 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).

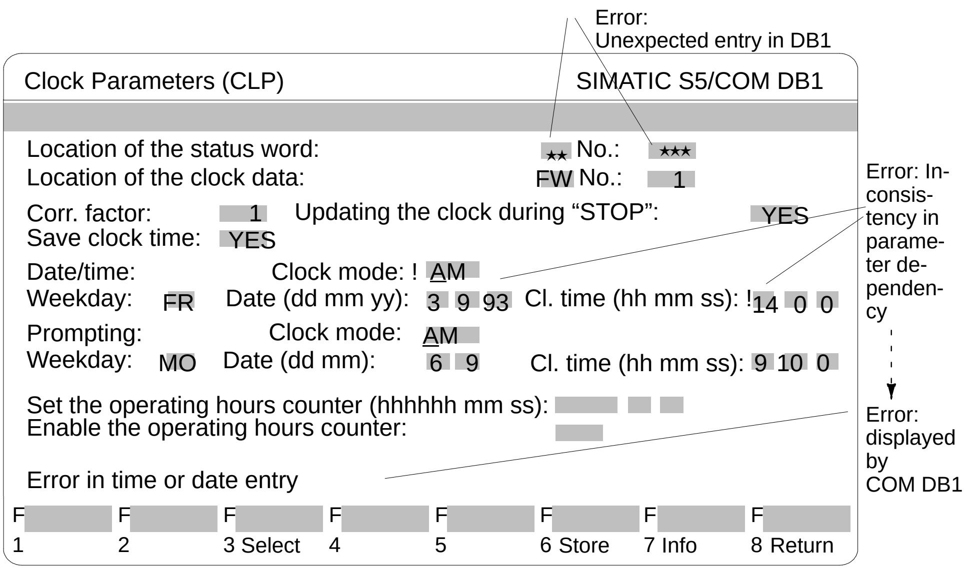


Figure 22-6 Display of Incorrect Parameters in the Parameter Assignment Dialog

Errors When Programming DB1
2

Illegal input is blocked by COM DB1 during programming:

- The input texts are checked by COM DB1 after the entry has been completed with the ***Return*** key.
  You are informed of syntax errors or value range violations with an **error message** e.g. `Invalid value range`). Incorrect 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 incorrect parameter settings found in this way are indicated with a ***!*** in front of the input field.

Note

If you position the cursor on the incorrect (***!***) parameter in the parameter assignment dialog, the relevant error message will appear in the message line.

Only after all parameters have been correctly entered can the parameter block be stored with ***F6*** *= Store.*

22.4 Example of a Complete DB1 Parameter Assignment with COM DB1

Overview

Based on a concrete example, this section shows you how to proceed when assigning parameters with COM DB1. This section is concerned with the handling of COM DB1 and not with the function requiring values in DB1.

You will find an explanation of the function and its parameters in the relevant PLC manual. The example below will familiarize you with handling COM DB1.

The table below contains:

- All the steps required to assign parameters to a PLC;
- All the dialogs 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 22-1 *Overview of Procedure for Assigning Parameters to a PLC with COM DB1*

Steps to be Executed in the Following Order and...	**Dialogs Required**
1. Install COM DB1	
2. Start COM DB1	
3. Select language	*Select Language* dialog
4. Enter defaults	*Defaults* dialog
5. Switch PLC from RUN to STOP	*PLC Functions* dialog
6. Load Default DB1 from PLC; Enter comment for DB1; Select parameter block	*Loading DB1* dialog
7. Enter comment for parameter block	*SINEC L2 Overview table* dialog
8. Edit parameters	*Basic Parameters* dialog *Standard Connection* dialog
9. Output DB1 to printer	*Printing DB1* dialog
10. Transfer DB1 to PLC	*Transferring DB1* dialog
11. Save DB1 to STEP 5 program file	*Transferring DB1* dialog
12. Switch PLC from STOP to RUN	*PLC Functions* dialog

Description of example task

An S5-95U with integral SINEC L2 interface is to be assigned parameters. The S5-95U will 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.)

Requirements

Please note the following requirments:

- An S5-95U with SINEC L2 interface (Order No.: 6ES5 095-8MB12, Version 01).
- A PG 7XX programming device plugged into the programming device port of the S5-95U.
- The bus connector must not be plugged into the SINEC L2 interface.
- The S5-95U must be in the *RUN* mode.
- You have created a program file *AG95L2ST.S5E* with the STEP 5 package.
- You have created a printer file or footer file with the STEP 5 package.

22.4.1 Preparations

Selecting the Language

After starting COM DB1, the *Language* menu appears. Use keys ***F1*** to ***F5*** to select the language in which COM DB1 will appear on the screen.

- Press ***F2*** = *English*. (You can exit COM DB1 by pressing ***F8*** = *Return* or the ***ESC*** key.)

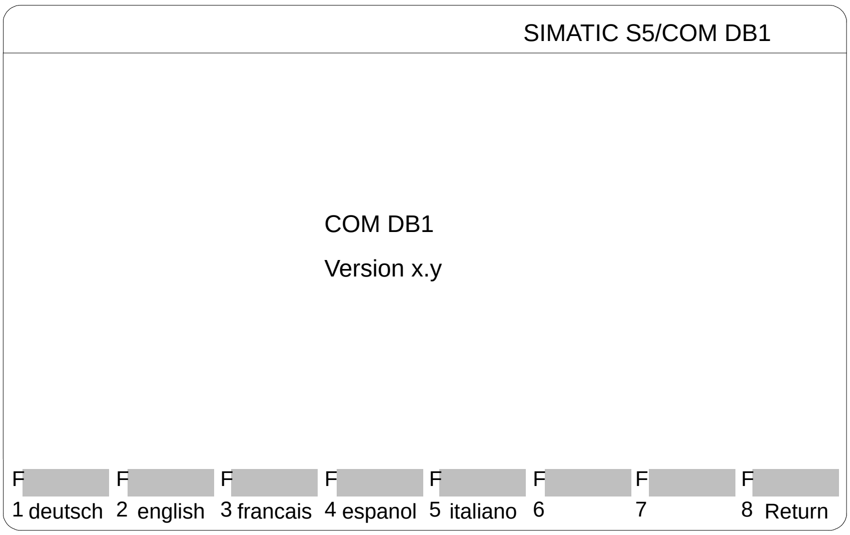


Figure 22-7 COM DB1 *Language* Dialog

Setting Defaults

You define the defaults for parameter assignment with COM DB1 in the *Defaults* dialog as described below.

Mode between COM DB1 and the CPU:

After selecting the Defaults dialog, the cursor is positioned in the *Online/Offline:* input field.

1. Press ***F3*** *= Select* to open the list box belonging to the *Online/Offline:* input field.
2. Press the ***Return*** or ***INSERT*** key to enter *Online* in the input field. *Online* appears in the input field.
3. Press the ***Return*** or ***INSERT*** key to position the cursor on the next input field.

Defining the Order Number

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 list box.)

Defining the PLC Revision Level

Enter PLC revision level *01* at the keyboard and complete the entry by pressing the ***Return*** or ***INSERT*** key. (You can cancel the entry with ***ESC***, i.e. the input field is empty again.)

When you have entered all defaults, the dialog appears as shown below:

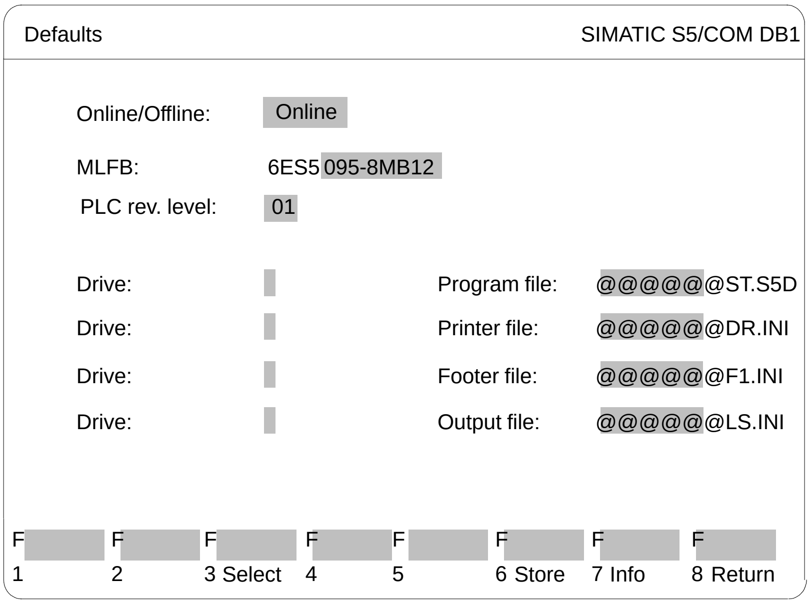


Figure 22-8 COM DB1 *Defaults* Dialog

Store the entries by pressing ***F6*** *= Store*. The *Overview table* dialog 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* dialog.

COM DB1 displays the following table 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

F1	F2	F3	F4	F5	F6	F7	F8
New DB1	Load DB1	Transfer DB1	Print DB1	Delete Block	Select Block	PLC Function	Return

Figure 22-9 COM DB1 *Overview Table*

Changing the Operating Mode

You can change the operating mode of the PLC in the *PLC Functions* dialog:

1. Press ***F7*** = *PLC function*.
2. Change the operating mode by pressing ***F2*** = *Run → Stop*. The PLC is now in the STOP mode.

PLC functions SIMATIC S5/COM DB1

F1	F2	F3	F4	F5	F6	F7	F8
Compress PLC	Run –> Stop	Stop –> Run					Return

Figure 22-10 COM DB1 *PLC Functions* Dialog

3. Press ***F8*** = *Return* to return to the *Overview table*.

22.4.2 Loading the Default DB1 from the PLC; Entering Comments for DB1; Selecting the Parameter Block

Loading and Modifying DB1

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* (*see Figure* 22-11).
   The *Loading DB1* dialog appears as shown below:

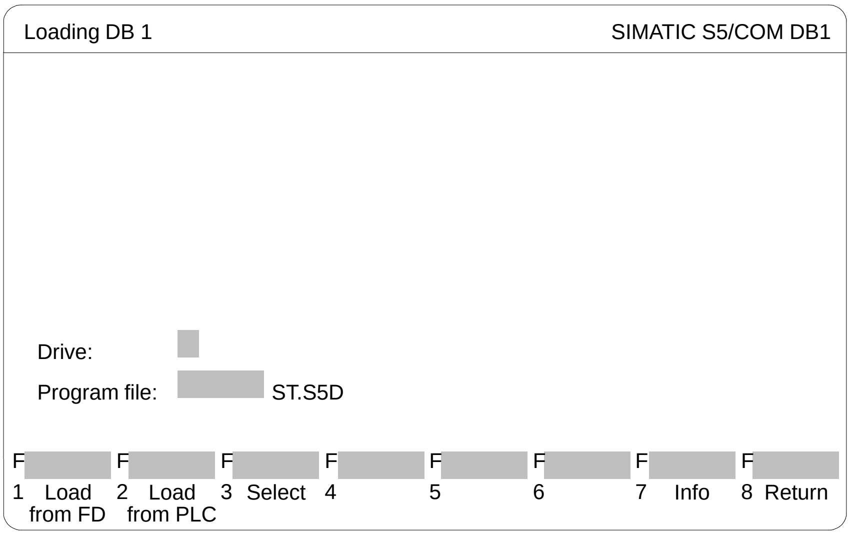


Figure 22-11 COM DB1 *Overview table* Dialog

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 (*see Figure* 22-11).

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* dialog.
2. Enter the comment, consisting of up to 80 characters; for our example: *Parameter assignment for SINEC L2 interface (standard connection only)*
   (→ Figure 22-12).
3. Press either the Return 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*.

Overview table SIMATIC S5/COM DB1

Parameter assignment for 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

F1	F2	F3	F4	F5	F6	F7	F8
New DB1	Load DB1	Transfer DB1	Print DB1	Delete block	Select block	PLC Function	Return

Figure 22-12 COM DB1 “Overview table” Dialog

2. Press either the ***Return*** or ***INSERT*** key. The *Overview table SINEC L2* dialog appears on the screen.

Entering Comments for the Parameter Block

You can enter a comment relating to the SINEC L2 parameter block in the *Overview table SINEC L2* dialog.

1. Press ***COM***. The cursor is now in the comment line.
2. Enter the comment consisting of up 80 characters; for our example: *Parameter assignment for standard connection between station 2 and station 1*.
3. Press either the ***Return*** or ***INSERT*** key. The cursor then appears in the line *Basic parameters*.

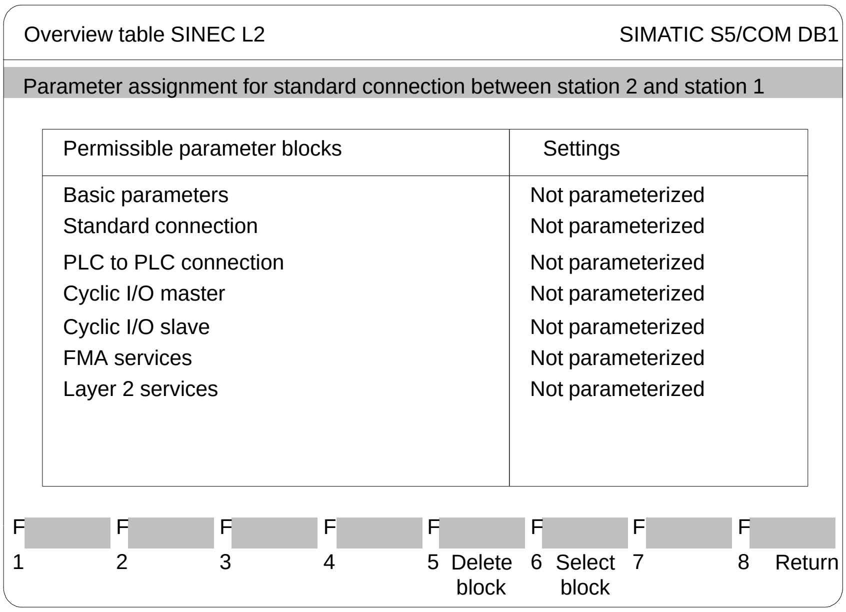


Figure 22-13 COM DB1 *Overview table SINEC L2* Dialog

Editing Parameters

In the *Overview table SINEC L2* dialog, 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* dialog, the cursor is positioned in the *Basic parameters* line.
2. Press either ***F6*** *= Store*, the ***Return*** or ***INSERT*** key. The *Basic parameters* dialog appears (*see Figure* 22-14).

Defining the station number

After selecting the *Basic parameters* dialog, the cursor is positioned in the *Station number:* input field.

1. Enter *2* at the keyboard.
2. Store the entry by pressing the ***Return*** or ***INSERT*** key. The cursor is now at the next input field. (you can cancel the entry with ***ESC***, i.e. the input field will be empty again.)

Defining station status:

1. Press ***F3*** *= Select* to open the list box belonging to the *Station status:* input field.
2. The cursor is at the *ACTIV(E)* line of the list box.
3. Enter *ACTIV(E)* in the input field by pressing the ***Return*** or ***INSERT*** key.
4. Position the cursor in the next input field by pressing the ***Return*** or ***INSERT*** key.
5. Enter all further arguments of the basic parameters as described above:

– Either direct at the keyboard (you can call up a display of the value range of the arguments with ***F7*** *= Info*) or

– Using the list box.

Refer to Figure 22-14 for the parameter arguments.

When you have entered all basic parameter arguments, the dialog 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

F1	F2	F3	F4	F5	F6	F7	F8
		Select			Store	Info	Return

Figure 22-14 COM DB1 *SINEC L2 Basic Parameters* Dialog

6. Press ***F6*** *= Store*. The basic parameters are stored in DB1 and the *Overview table SINEC L2* dialog appears (*see Figure 22-14*). *Parameterized* appears in the *Basic parameters* line in the dialog.

   (Press ***ESC*** or ***F8*** *= Return* to cancel the entry. The *Overview table SINEC L2* dialog then appears in its original form.)

Editing Parameters for Standard Connection

Select Standard connection*:*

The cursor is in the *Overview table SINEC L2* (*see Figure 22-15*) dialog in the *Standard connection* line.

1. Press either ***F6*** *= Store*, the ***Return*** or ***INSERT*** key. The *SINEC L2 Standard Connection* dialog appears.
2. Enter all parameter arguments as described for the basic parameters either directly at the keyboard or using the list box.

Refer to *Figure 22-15* for the parameter arguments.

When you have entered all the arguments, the dialog 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	

F 1	F 2	F 3 Select	F 4	F 5	F 6 Store	F 7 Info	F 8 Return

Figure 22-15 COM DB1 *Standard Connection* Dialog

3. Press ***F6*** *= Store*. The parameters are stored in DB1 and the *Overview table SINEC L2* dialog appears (*see Figure 22-15*). *Parameterized* appears in the *Standard connection* line in the dialog.

   (Press ***ESC*** or ***F8*** *= Return* to cancel the entry. The *Overview table SINEC L2* dialog 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 created.

1. Press ***F8*** = *Return* twice to return to the *Overview table* dialog.

The *Overview table* dialog has changed; the SINEC L2 parameter block is displayed as having parameters assigned:

Overview table SIMATIC S5/COM DB1

Parameter assignment for 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

F1	F2	F3	F4	F5	F6	F7	F8
New DB1	Load DB1	Transfer DB1	Print DB1	Delete block	Select block	PLC function	Return

Figure 22-16 COM DB1 *Overview table* Dialog

2. Press ***F4*** = *Print DB1*. The *Printing DB1* dialog appears as shown below:

Printing DB1 SIMATIC S5/COM DB1

F1	F2	F3	F4	F5	F6	F7	F8
Print printer	Print on FD						Return

Figure 22-17 COM DB1 *Printing DB1* Dialog

3. Press ***F1*** *= Print printer.*
   This prints the “Overview table” dialog, the *Overview table SINEC L2* dialog and all parameter assignment dialogs 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* dialog automatically appears.

(If DB1 has not been printed, you will receive a relevant message.)

Transferring DB1 to the PLC

You want to transfer the DB1 you have just created to the PLC.

1. Press ***F3*** *= Transfer DB1* in the *Overview table* dialog (*see Figure 22-18*).

The *Transferring DB1* dialog appears as shown below:

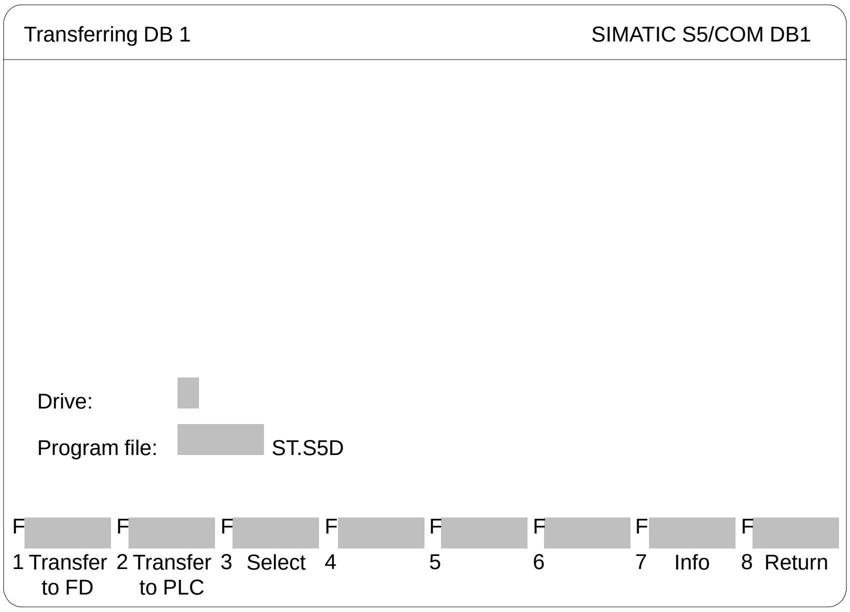


Figure 22-18 COM DB1 “Transferring DB1” Dialog

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* dialog automatically appears. (If there are errors in DB1, a message is displayed and DB1 is not transferred.) The incorrect parameter block is indicated in the Overview dialog.

Saving DB1 in a STEP 5 Program File

You want to save the DB1 you have just transferred to the PLC in a STEP 5 program file (or on diskette). You must specify the STEP 5 program file ino which DB1 will be stored in the *Transferring DB1* dialog. It was a requirement for our example that you had already created the STEP 5 program file *AG95L2ST.S5E* with the STEP 5 package.

1. Press ***F3*** *= Transfer DB1* in the *Overview table* dialog (*see Figure 22-19*). The *Transferring DB1* dialog appears.
2. Enter the STEP 5 program file and the drive (*see Figure 22-19*).

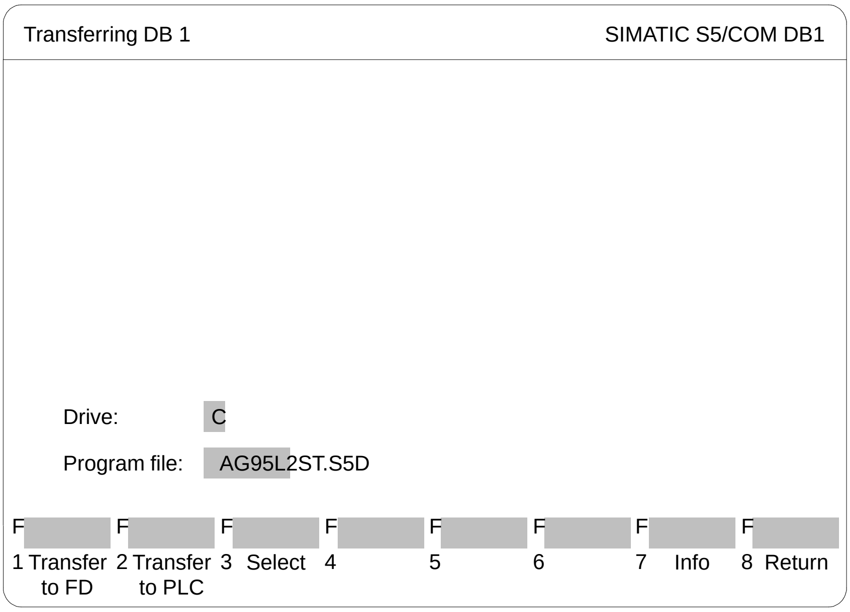


Figure 22-19 COM DB1 *Transferring DB1* Dialog

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* dialog automatically appears.(If there are errors in DB1 a message is displayed and DB1 is not transferred.) The incorrect parameter block will be indicated in the Overview dialog.

Switching the PLC from STOP to RUN

You can change the operating mode of the PLC in the *PLC functions* dialog.

1. Press ***F7*** *= PLC functions* in the *Overview table* dialog (*see Figure 22-20*). The *PLC functions* dialog 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 the ***Return*** or ***INSERT*** key. The parameter settings will be transferred to the operating system of the PLC.

   (You can cancel updating in the PLC with ***ESC*** or ***F8*** *= Return*.)

The parameter settings in the PLC have been updated and the PLC is in RUN.

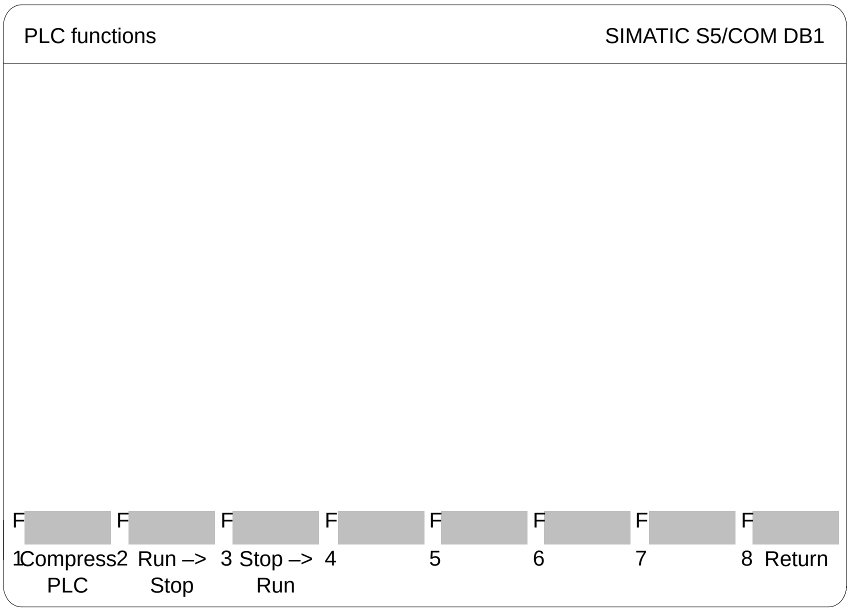


Figure 22-20 COM DB1 *PLC Functions* Dialog

4. Exit COM DB1 by pressing ***F8*** *= Return* 4 times.

PG Link

23

Overview

The task of the PG link package is the exchange of STEP 5 blocks or files between various programming devices.

Chapter Overview

23.1 Hardware

Hardware Requirements

Data exchange with the partner PG is only possible via an **active** TTY port (20 mA). If the existing COM1 port is only equipped with a V.24 or passive TTY port, 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 port of the PG to an active TTY port and therefore simulates the S5 interface of an S5 programming device.

You connect your PG with the partner in one of the two following ways:

- Via the **active TTY port COM 1**

  The PG and the partner PG are connected via two connecting cables.

or

- Via the **passive TTY or V.24 port COM 1**

  If you have a PG with a passive TTY port or with only one V.24 port COM 1 the passive port 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.

23.2 Linking

Loading PG Link

You load this package with the menu command **Change ▸ Others ... F9**. The PG Link program is in the directory ...\S5_SYS\S5_COM\PG_PG

As soon as you activate the PG Link package, it is started and you change to the user interface of the PG Link package.

PG Link

The PG Link package provides the following functions:

- Switching the PG to passive. For data exchange, a passive and an active PG are required.
- Sending data from the active to the passive PG
- Fetching data from the passiven to the active PG

Selecting Your Settings

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 field 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 ***Insert*** 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 entries:

Key level		**Effect of the function keys**
1	**2**	
F1		PASSIVE This switches the programming device from the ACTIVE to PASSIVE status. The PG to which data are sent must always be PASSIVE. The passive setting is canceled 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.
		Example **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 ***Insert*** key and the transfer to the partner PG begins automatically.

Key level 1	Key level 2	Effect of the function keys
	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 ***Insert*** key and the transfer to the partner PG begins automatically.
	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 ***Insert*** 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 ***Insert*** 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

Part 5: Practical Example

Practical Application of STEP 5
– Programming Example –

24

Practical Application of STEP 5 - Programming Example -

24

Overview

To help you get to know STEP 5 and get used to working with this software package, this Chapter contains a sample application. The control task *controlling a car wash* shows you step by step how to edit, test, document and archive a user program.

Chapter Overview

24.1 Introduction to the Example (Control Task)

Overview

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 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 *Section 24.5.4 Designing a Program for the Sample Application*, 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 startup and cyclic operation of the car wash.

It is advisable to try out the steps explained in *Section 24.2* on your PG. You will probably only need to edit a few segments. You will find the complete function block along with all the other parts of the example program in the directory:

C:\STEP5\S5_SYS\EXAMPLE .

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.

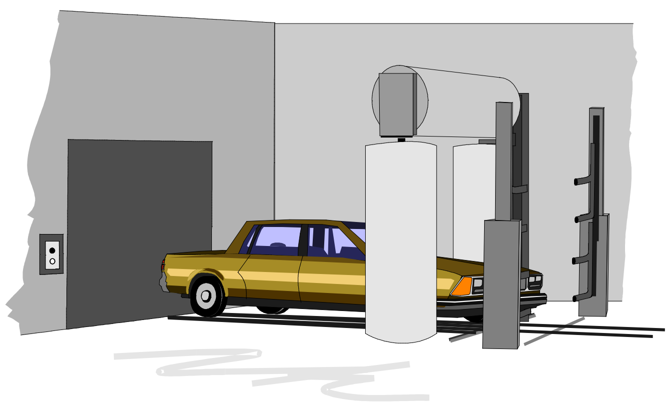


Figure 24-1 Carwash

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* 24-2). By labelling the I/O signals based on the verbal description of the process, the control program to implement this process can be developed.

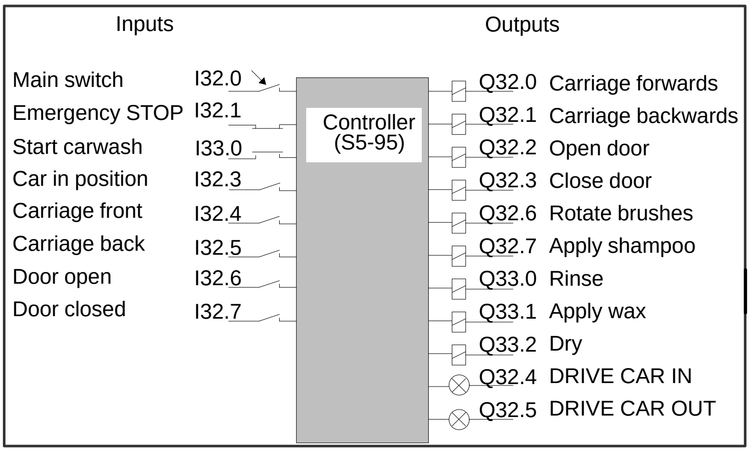


Figure 24-2 Controller with Process Interfaces

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.

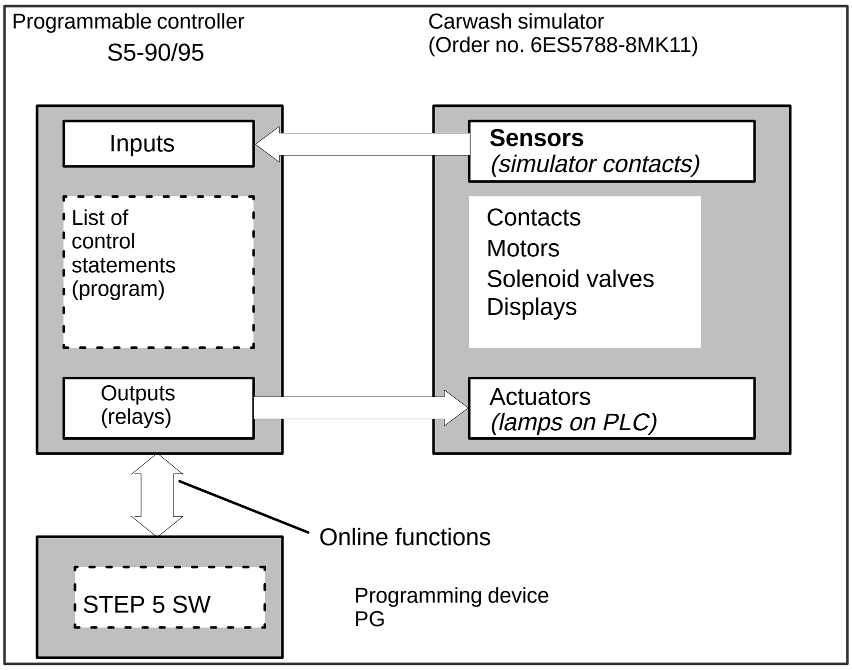


Figure 24-3 Configuration of the *Carwash* Example

24.2 Creating a 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.

24.2.1 Setting up the Project

Since the operating system and the programmer startup 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.

File
```
Project >
  Set       F4
```

Beginning with the menu selection **File ▸ Project ▸ Set F4** you make all the settings and parameter assignments necessary to prepare for programming.

1. For a new project, you select **Project ▸ Set**. To select the existing project at a later date, you use **Project ▸ Load**. The six tabs for the project settings appear. Here you select the files you require for your project. These files either have defaults or *NONAME* entered.

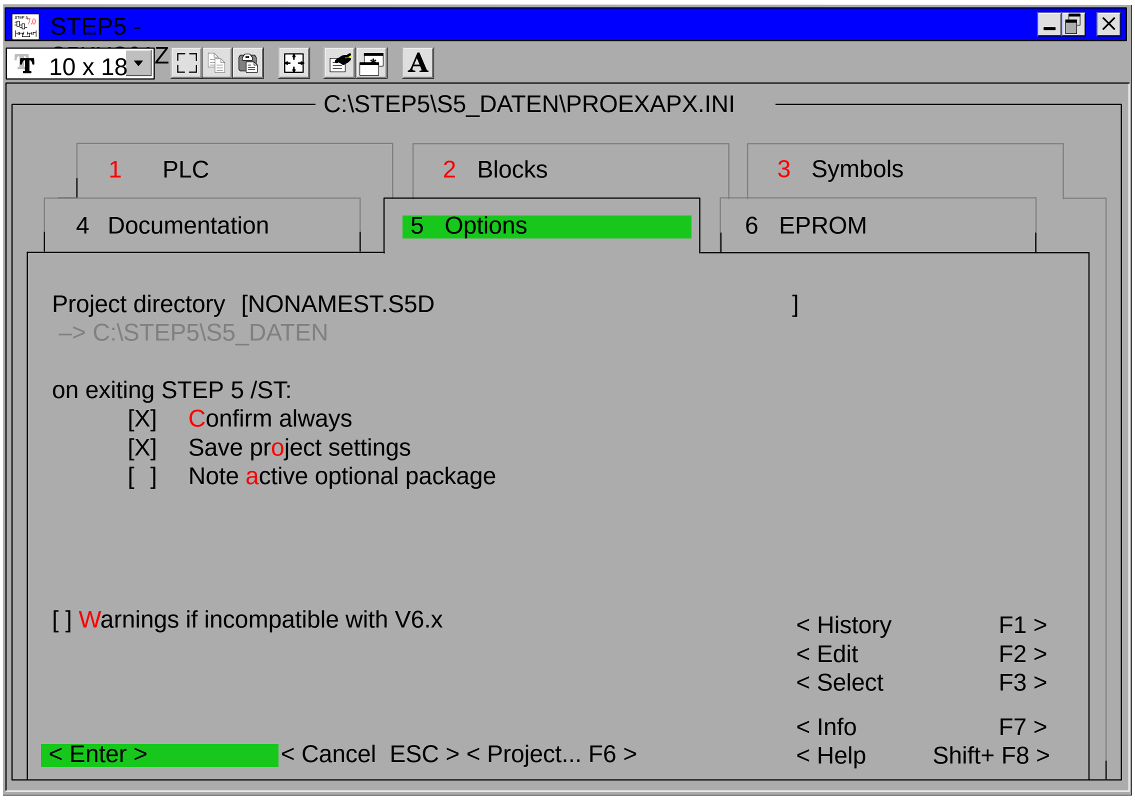


Figure 24-4

Making Settings in the Tab Pages

Make all the settings to prepare for programming the carwash project as follows:

1. Name the program you want to create for the carwash by entering the project directory in tab page 5 *Options* with the following name:

   `C:\STEP5\S5_Daten`

2. In tab page 1 *PLC* select the mode. As long as there is no PLC connected, only *offline* is possible as the mode and this is preset by STEP 5.

3. In tab page 2 *Blocks* select the program file:

   `C:\CARWASST.S5D`

   Since we want to program in *Statement List*, set the parameter *Representation* to STL by pressing ***F3***.

4. Select the symbols file in tab page 3 *Symbols*:

   `C:\CARWASZ0.INI`

   By clicking [X] or pressing the ***F3*** key, the parameter *Display symbolic* is set.

   To simplify matters, we will leave the maximum symbol length at 8 characters. Since, however, a more detailed explanation will be helpful, change the comment length to 40 characters. You must complete this entry with the ***Return*** key.

5. Select a printer file (*DR.INI) in tab page 4 *Documentation* or overwrite the default NONAME.

Save the Settings

File

```
Project >
  Save As...
```

You return to the menu by clicking ***Enter***.
After selecting **Project ▸ Save As**...*;* the file Save project settings dialog appears in which you enter *CARWAS* as the project file name.

After clicking ***Save*** and acknowledging the message `Destination file already on FD, overwrite?`, STEP 5 sets up the project file *CARWAS***PX.INI,** which contains the program files and settings.

24.2.2 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:\STEP 5\S5_SYS\EXAMPLE under the project name PROEXAPX.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 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 24-1*), 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. Start the *assignment list* STEP 5 editor in the editor menu (or press function key ***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 24-5 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*** = *Enter.*

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 24-5 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 **one** error is found, the incorrect line is displayed at the top.

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

1. Press ***OK*** and ***Continue***.

- This brings you in the initial menu.

Management
`Assignment List..`

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 start the translation again.

Editing the Data block

Editor
`Data Block`

1. You call the editor for creating data blocks in the menu under **Editor ▸ Data block in the program file ....** (or function key ***F2***). Use *Figure 24-14* for the contents of the data block.
2. Enter the type and number of the data block to be created in the job box, in this case: DB5. Confirm this with ***Edit***.

   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.
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 `Illegal operation`.
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 ***Return*** key.
5. The next DW field (following line) is displayed with the same format. If you require a different format 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 24-14*.

Correcting in the Data Field

Function	Setting
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.

- 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*

1. Type in the text after pressing ***SHIFT F6*** or ***COM****.*
2. 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.

- Type in the text from *Figure 24-14*, completing each line with the ***Return*** *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.

1. Type in *Simatic–*. The text is inserted at this point. You return to the overwrite mode with ***F1*** *= Overwrite.*
2. 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).

3. 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:

1. Complete editing the DB by pressing the ***Insert*** key.
2. 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`

1. You call the editor for creating STEP 5 blocks in the menu **Editor** ▸ ***STEP 5 Block F1***. 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:

1. Press ***F3*** *= Select.*
2. 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.
3. Mark the options

- *Confirm before overwriting* and
- *Update assignment list*

with ***F3*** and then close the box with ***Edit****.*

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.

1. 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.

2. 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 of Section 24.5.4.

1. 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.

2. 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.

1. Press ***COM*** and ***SHIFT F6*** *= Title*
2. 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.

1. Press ***Seg End*** *(***)*

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

2. Type in the statements and statement comments based on printed program excerpt. 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.

1. 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.

2. 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 (only 1 cycle!)*.

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

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

Start the segment comment editor as follows:

1. 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 A).

2. 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.

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

Inserting Segment 3

1. 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.

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

24.2.3 Documenting the Program

Documentation
`STEP 5 Blocks...`

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.

Please proceed as follows:

1. Enter *FB 5* from your program file in the job box.
2. Under the options, select the STL address representation and the printout type *standard*.
3. The printout is triggered with ***Output***.

The printout contains the following elements for each segment:

- the segment title and segment number
- the statement 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 (step 5) shown in Section 24.5.4 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.

Output to a File

You can output the documentation to a file.

- In this case, mark *Output to file* in the job box and specify the file name *CARWASLS.INI*.

24.3 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.

File
```
DOS File
  Copy
```

The complete program is located under the name PROEXA... in the directory C:\STEP 5\S5_SYS\EXAMPLE. To transfer the file, change over to the DOS file functions as follows:

1. Select ***DOS File*** and ***Copy*** in the ***File*** menu.

   The job box *Copy DOS files* is displayed. Here, you select the source and destination directory for the transfer.

2. First check that the directories are correctly selected.

   Source drive: `C:\STEP 5\S5_SYS\EXAMPLE`

   Destination drive: `C:\STEP 5\S5_DATEN`

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

1. Mark *all* in the *Copy mode* window and select *yes* in the *Confirm before overwriting* window.
2. Trigger the transfer by clicking on ***Copy*** 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.

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

File
```
DOS Files
  Directory
```

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

- set the directory C:\STEP 5\S5_DATEN under *Dr/directory.*

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

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

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... .

File
```
Blocks >
  Transfer
```

To transfer blocks, select **Blocks ▸ Transfer**, STEP 5 then displays a job box in which you specify the following:

1. under *Transfer from* you specify the program file PROEXAST.S5D and under *transfer to* the program file CARWASST.S5D

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

2. In the job box *Transfer blocks select* the field *Block List* and enter *FB10.*

   After clicking on ***Transfe*r** or pressing the ***Return*** key, STEP 5 displays the prompt *Transfer comments as well?*.

3. 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.

---

4. Nach dem Kopiervorgang exit the job box with ***ESC*** *= Exit.*

File
```
Blocks >
  Directory > F3
```

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

1. Select the menu **File ▸ Blocks ▸ Directory** or use ***F3*** in the selection box *Block - Directory:Settings*.
2. Enter A in the block list.
3. 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.

Editor
```
STEP 5
Block...F1
```

To transfer segments

4. Go into the block editor and select the function block FB10 in the Edit STEP 5 block(s) job box.
5. Move the cursor to segment 6 using ↓ *= scroll down* or the ***+*** key.
6. Press ***F5*** *= Seg Fct* and ***F4*** *= File.*
7. 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

1. Select FB5 in the block editor and move the cursor to segment 5 at the end of the program.
2. Press ***F5*** *= Seg Fct* and ***F6*** *= Append.* Then press ***F2*** *= Buffer* to append segment 6 to the program CARWAS.... .
3. Complete the operation with ***F8*** *= Return* and ***F7*** *= Enter.* 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:

1. Select **Project ▸ Set.**
2. Enter *PROEXAZ**0.INI*** as the symbols file.
3. 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.

Transferring and Renaming Blocks

File

```
Block >
  Delete >
```

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 **File ▸ Blocks ▸ Delete**
2. Enter *FB5* in the *Block list* field.
3. After you press ***Delete****,* the system prompts `Delete comments as well?`.
4. Confirm this prompt with *Yes* and the message `Block deleted!` with *yes.*

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

File

```
Blocks >
  Transfer... F5
```

1. To rename FB10, select **File** ▸ **Blocks** ▸ **Transfer** and then enter or select the following:

- *Transfer from* C:CARWASST.S5D
- *to* C:CARWASST.S5D
- mark (X) *block* [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.

Transferring the Organization Blocks

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

File

```
Blocks >
  Transfer...F5
```

1. To transfer the OBs, select **File** ▸ **Blocks** ▸ **Transfer** and
2. enter in the job box

- *transfer from* PROEXA...
- and *to* CARWAS...
- *Mark Selection* (X) Block list and enter OB1, OB20, OB21, OB 22.

3. When you click on ***Transfer,*** the system displays the message `Transfer comments as well` and then `Blocks transferred`, which you 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.

24.4 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 by selecting *Generate XRF* in the *management* menu.

Management
`Generate XRF`

With the menu command **File** ▸ **Project** ▸ **Set** (Blocks tab), you can enter the cross reference list file CXR.INI. 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 for which the cross references will be displayed.
3. Confirm with the ***Return*** key.

A table of cross references for the selected operand is now displayed (*Figure* 24-6). 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 directly from OB20:

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 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

Cross references			
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 24-6 References to the Operand -STARTUP in CARWAS

Documentation

`Cross References`

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 24-7 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 XREF 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*

   S e a r c h  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 24-7 List of Cross References from the Carwash Program

Search

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

Editor
`STEP 5 Block...`

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

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.

Management
`Manual Rewiring`

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 ***Rewire****.*

- 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.*
5. Check that the modification has been made as follows: Call block FB10 in the editor and press ***F3*** *= Search, t*ype 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... that was modified by the rewiring function with the original FB in PROEXA...

File

```
Blocks>
  Compare...
```

1. Select **File ▸ Blocks ▸ Compare**.
2. In the job box, enter C:PROEXAST.S5D under *compare with program file* and FB10 under block list.
3. When you have done this, click on ***Compare****.*

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.

4. Repeat the block comparison by marking *all blocks* (A) in the job box.

STEP 5 displays the comparisons as shown in *Figure* 24-8. 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 24-8 Block Comparison between CARWAS and PROXEA

24.5 Loading and Testing the Program

To test the carwash program, you must now connect an S5-90/95 to your programmer. Establish the permanent connection between the PG and PLC as follows: change the mode to *online Modifiable [cycl.]* using ***F3** = Select* and ***Enter**.*

24.5.1 Loading the Program

File

```
Blocks>
  Transfer   F5
```

Load the program as follows:

1. Select **File** ▸ **BLocks** ▸ **Transfer**
2. If it is not already set, enter C:CARWASST.S5D as the source in the job box *Transfer from*
3. Select to **PLC**.
4. Under *selection, block list* enter *FB5* in the block list, then *DB5* and finally *all OBs*.
5. After pressing ***Transfer***, the blocks are copied to the PLC. Confirm this with *yes.*

File

```
Blocks>
  Directory...F3
```

1. Check the loading by outputting a list of the blocks on the PLC.
2. To do this, once again mark *all blocks* in the job box (A).
3. Select Transfer from **PLC**.

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.

24.5.2 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 24-35) 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.

24.5.3 Block Status

Test

```
Block Status
...Shift F6
```

1. On the simulator, switch all the toggle switches down (= off) and set the mode selector on the PLC to STOP.
2. Select **Test** ▸ **Block Status**.
3. Enter FB5 in the job box, mark the options with *yes* and click ***Output***.

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.

4. Now switch the PLC to RUN.

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

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

The displays 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.

8. Now move the breakpoint to segment 5, in which 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.

9. 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.

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

11. 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 brushe*s and *carriage forwards* are started (variable = 1). The step counter switches to 3.

12. Simulate the remaining parts of the washing process by changing the inputs according to *Table* 24–2 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).

13. 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.

Corrections

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****.*

File

```
Blocks>
  Transfer...F5
```

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,

1. Select **File ▸ Blocks ▸ Transfer**
2. Enter FB5 in the job box.
3. *Transfer from* (X) PLC
4. Select to program file (X)

Test

```
Status Variable
Shift + F7
```

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 24-9*.
4. Press ***F2*** = *Save_as* 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 24-9 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:

1. 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.

2. 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.

3. 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.

4. 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* 24-10. The brush carriage now moves forwards with the brushes rotating and the shampoo jets open.

5. 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.

6. 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 **Test ▸ Force variables** 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*.

1. 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).

2. 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:

3. 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
```

Figure 24-10 Operands and Signal States

24.5.4 Designing a Program for the Sample Application

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.

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).

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 Figure 24-11 must also be added to this list.

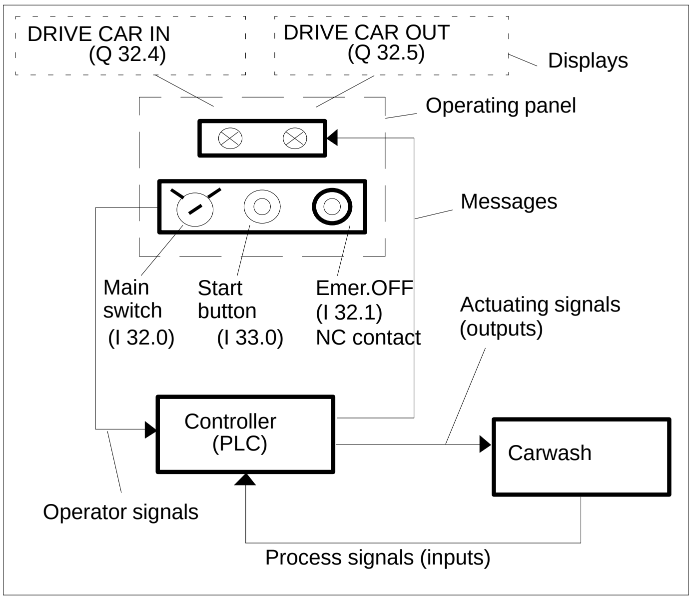


Figure 24-11 Control Structure with Operator Inputs/Outputs

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 24-1 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 24-2) 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 next parts of the washing process automatic without the driver taking any further action.

1. 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.

2. 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.

3. 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).

4. 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).

5. 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.

6. 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).

7. 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).

8. The carwash is empty.

   If there is no car in position (I **32.3** = 0) the system switches off the display DRIVE CAR OUT (**Q 32.5** = 0) and resets the 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 (*Figure 24-12*) 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 *Figure 24-12* .

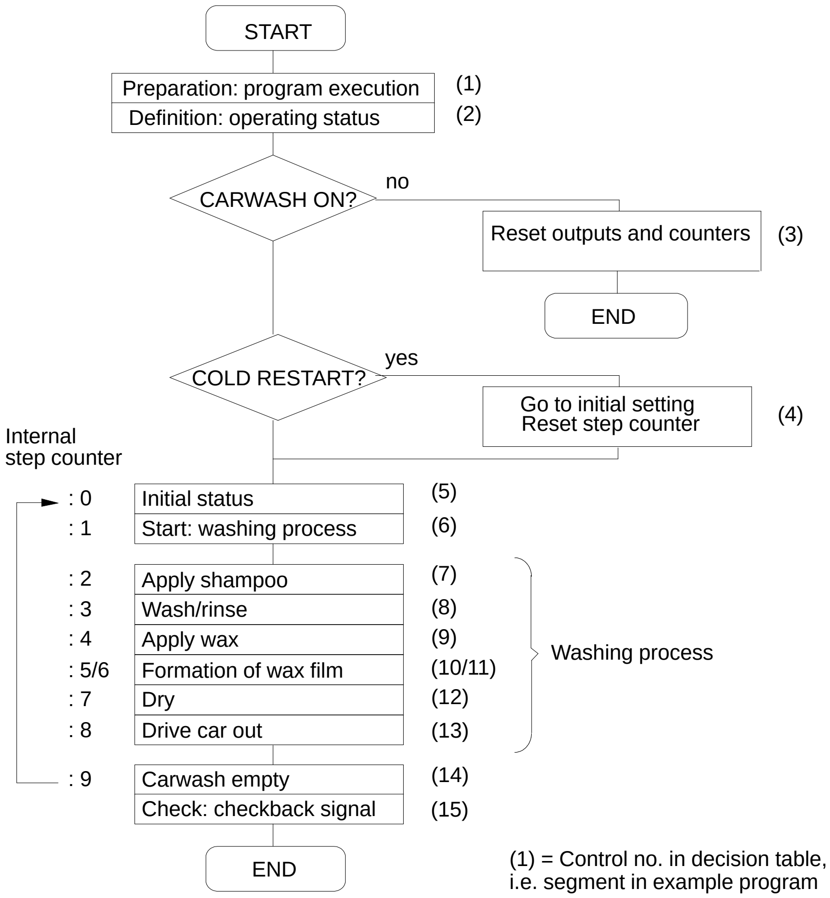


Figure 24-12 Flowchart of a Carwash Process

Table 24-2 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, Figure 24-13 illustrates the program structure with the block names as they are used in the example.

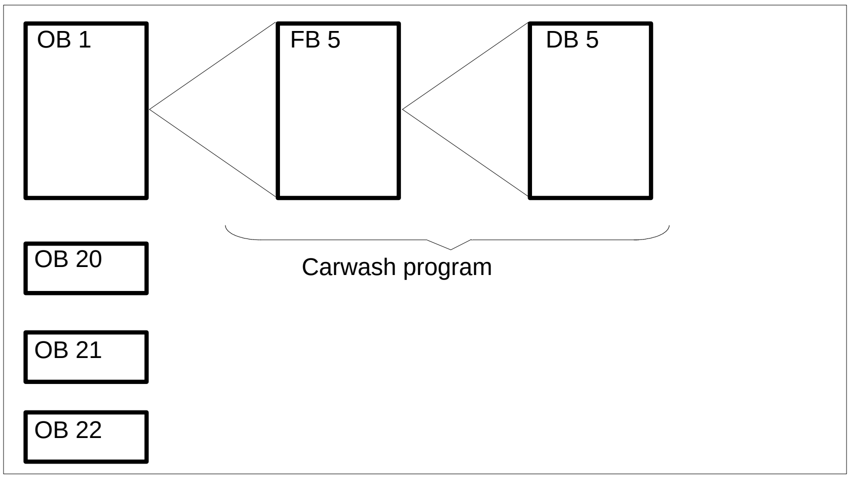


Figure 24-13 Structure of the Carwash Program

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 (*Figure 24-14*). 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:	K =	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 24-14 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   -MAINSWIT     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   -MAINSWIT     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   -MAINSWIT     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.

Part 6:
Data Management

STEP 5 Data Management 25

Overview

This chapter describes the structure of the user memory and how it is distributed in STEP 5. Tables show you which directories contain the files important to STEP 5. The product information contains detailed information about the directories and files on your device.

Chapter Overview

25.1 RAM Memory Requirements for STEP 5/ST V 7.0

Overview

To allow STEP 5/ST V 7.0 to operate with all its functions in conventional memory, a free RAM memory capacity of at least **550 Kbytes** is required after you have loaded the operating system.

The management of the user memory is already optimized on a PG supplied with STEP 5.

If you install STEP 5 later or if you change the configuration of your system or load other drivers or programs, it may be necessary to change the assignment of the user memory to avoid errors.

Memory Configuration

The memory configuration and management can affect the following:

- Which programs can be run
- How fast programs can be run
- How much data a program can work with
- How much data can be stored between one working session and the next.

User Memory

The basic configuration of the user memory is on the mother board of your programming device. This can be extended by a memory expansion card. All programs must be loaded in the user memory before they can be run.

Your PG has two different types of user memory:

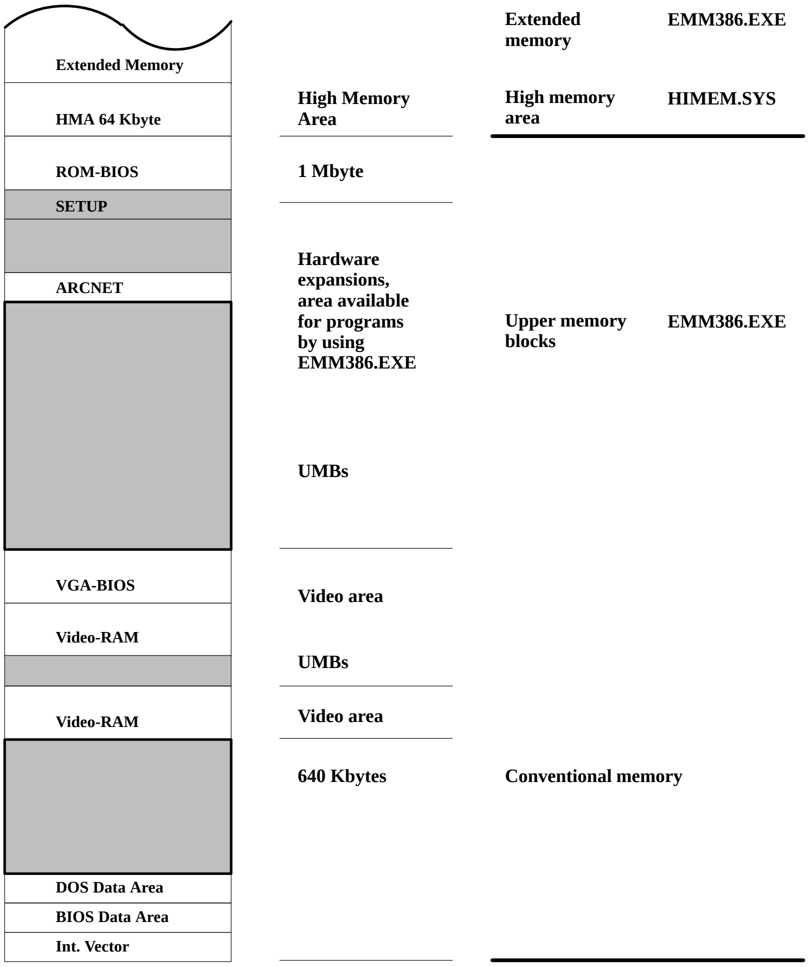
- Conventional memory
- Extended memory.

Programs running under MS-DOS normally use the PG’s conventional memory. To allow programs to use the extended memory, you must install a memory manager to coordinate access to this memory.

25.2 Memory Distribution

Example

The diagram illustrates an example of memory distribution:



Conventional Memory

The conventional memory has a standard capacity of 640 Kbytes in all PGs. Programs can use the conventional memory without the special commands required for other types of memory.

MS-DOS occupies part of the conventional memory. The device drivers and commands specified in the CONFIG.SYS and AUTOEXEC.BAT files take up further user memory space. The remaining memory is available for user programs.

Upper Memory Blocks

In addition to the conventional memory, your PG also has a 384 Kbyte memory known as the *upper memory blocks*. This area is immediately after the 640 Kbytes of the conventional memory. This area is normally reserved for your additional hardware, however parts of it can be made available by a memory manager.

High Memory Area

The **h**igh **m**emory **a**rea (HMA) is a special 64 Kbyte field in the additional memory located directly above the 1 Mbyte address.

Extended Memory, XMS

Most programs use the conventional memory. They cannot use the extended memory because the *addresses* which identify the locations of the programs in the extended memory are too high for these programs to recognize. Only the addresses in the 640 Kbyte area of the conventional memory are recognized by all programs.

You can activate more user memory in your programming device by installing a memory manager. These programs allow access to the extended memory and the upper memory blocks.

25.2.1 MS-DOS Memory Manager

A memory manager is a device driver that allows access to or manages certain types of memory.

MS-DOS (5.0 and 6.2) has the following installable memory managers:

- HIMEM.SYS manages access to the extended memory
- EMM386 allows access to the extended memory. It also allows access to the **u**pper **m**emory **b**locks (UMBs).

To install a memory manager, use the **DEVICE** command in your CONFIG.SYS file. Although memory managers occupy a part of the conventional memory, they make up for this by allowing access to far greater areas of memory in the extended memory or upper memory area than they themselves occupy.

Running MS-DOS in the High Memory Area

MS-DOS is usually run in the conventional memory. This restricts the conventional memory available for user programs. MS-DOS can also be run in the extended memory. In this case, it uses the 64 Kbytes of the *high memory area* (HMA). Since few programs use this area it may prove useful to run MS-DOS here.

Running MS-DOS in the extended memory area has the following advantages:

- Approximately 40 Kbytes of conventional memory are released
- It uses the high memory area, part of the extended memory used by very few programs.

The command **DOS=HIGH,UMB** specifies the area of the user memory in which MS-DOS will be located and determines whether or not upper memory blocks will be used.

Using the Upper Memory Blocks

Another way to gain more memory over and above the 640 Kbytes of user memory is to install the memory manager EMM386.EXE.

The memory manager can make available part of the extended memory area from 640 Kbytes to 1 Mbyte reserved for hardware. These parts are known as the *upper memory blocks* or UMBs.

Use: with the command **DEVICEHIGH** <driver file> in the CONFIG.SYS file, you load drivers in the upper memory blocks.

Setting up a Larger User Memory

Even when your memory capacity is adequate, you may not be able to run a program. Memory-resident programs often occupy part of the user memory so that there is not enough user memory free.

Normally this results from having too little conventional memory.

In this situation, making use of HIMEM.SYS has the following advantages:

- It makes the extended memory available to programs which use this memory according to XMS (the E**X**tended **M**emory **S**pecification).
- It prevents system errors caused by programs with contradictory memory requirements.
- It allows you to run MS-DOS in the high memory area of the extended memory.
- It allows EMM386 to use the extended memory.
- It allows the use of the upper memory blocks (UMBs) in conjunction with EMM386.EXE.

Order of the Drivers

The order of the drivers in your CONFIG.SYS file can be important. It can have an effect on the rational utilization of memory and the problem-free running of various programs.

The following list shows the order in which you should load device drivers in your CONFIG.SYS file (with the command `DEVICE` or `DEVICEHIGH`):

1. `HIMEM.SYS`.

   Example:
   `DEVICE=C:\DOS\HIMEM:SYS /M:1`

   The option /M:1 stipulates the ROM-BIOS used.

   The driver `HIMEM.SYS` should be the first driver to be loaded in `CONFIG.SYS`.

2. EMM386.EXE

   Example:
   ```
   DEVICE=C:\DOS\EMM386.EXE RAM I=B000-B7FF I=C800-DFFF
   X=E000-E0FF I=E100-F5FF FRAME=D000
   ```

   This command loads the MS-DOS memory manager EMM386.EXE from the \DOS directory in the user memory. It manages the extended memory and the upper memory area.

3. All device drivers which use the extended memory.

To keep as much conventional memory free as possible and to increase the functionality of STEP 5/ST, do not load any drivers that will not be used.

Parameters

RAM	This parameter provides you with an EMS window.
FRAME	This parameter indicates the place in the memory at which the EMS window should be located.
I=B000-B7FF	This 32-Kbyte area is normally intended for the Hercules software video interface. Since this area is not occupied on your programming device, it can be used as user memory.
I=C800-DFFF	When your programming device is shipped, this area is not occupied by hardware. It can therefore be used as user memory.
X=E000-E0FF	If this area is occupied by hardware, it must be excluded (not for the PG 740 and PG 760).
I=E100-F5FF	The area for the SETUP program can be used since EMM386 activates the protected mode and SETUP cannot be run in this mode (not for the PG 740 and PG 760).
Explanation:	I = **I**nclude, X = E**x**clude

25.2.2 Optimizing Hard Disk Access

SMARTDRIVE is an optimizer program that uses part of the extended memory to accelerate hard disk access.

```
DEVICEHIGH=C:\DOS\SMARTDRV.SYS 2048 /X
```

The above command loads SMARTDRV.SYS in the upper memory area of the user memory above the 640 Kbyte boundary. The number 2048 stipulates the maximum size of the cache as 2048 Kbytes. Values between 128 Kbytes and 8182 Kbytes (8 Mbytes) can be selected.

25.3 STEP 5 Directory Structure

STEP 5/ST uses a strictly defined directory structure. The structure consists of 4 separate directories.

`System directory`	All the files required for running the program are installed here. The user must not make any changes whatsoever within the system directory. The complete system directory can be made read-only (exception: subdirectory `S5_COM\...` must not be set to READ-ONLY).
`Home directory`	The files modified by the user are stored here. These include the batch files for starting the program, various INI files describing the workstation (device-specific data) and printer parameter files and path files modified by the user (DR.INI and AP.INI).
`SINEC`	The shipped MS–DOS drivers for SINEC L2 or H1 (SIMATIC NET network drivers) are copied to this directory. This directory is on C: and you cannot select a different drive for it.
`S5_INFO`	The product information and readme files are copied to this file. This directory is on C: and you cannot select a different drive for it.

The system and home directories can be located on different drives. You can also select the directory name you wish to use during installation.

`dr:\system_directory\S5_SYS\`		All STEP 5/ST programs and system files
	`S5_COM\`	Com adapter for V5 & V6 COM packages (this subdirectory must not be RO!)
`S5.COM\`	`AWLBAT\`	STL Batch Compiler
`S5.COM\`	`COM_DB1\`	COM DB 1
`S5.COM\`	`PG_PG\`	PG-PG Link
	`EXAMPLE\`	STEP 5 sample programs
	`DR_INI\`	Shipped printer parameter files (*DR.INI)
	`AP_INI\`	Shipped path files (*AP.INI)
	`S5_INST\`	Installation components

The part of the path shown in lower case letters:

`dr:\system_directory`

can be selected by the user during installation. All the directory names in upper case letters are created automatically with these fixed names.

Home Directory

`dr:\home_directory\S5_HOME\`	Device-specific data

User Directory

`lw:\home_directory\S5_DATEN\`	This is the default directory for user data after STEP 5/ST is installed. After initial installation, the directory is empty.

The parts of the path shown in lower case characters (dr:\home_directory) can be selected by the user during installation. All the directory names shown in upper case letters are created automatically with fixed names.

During the standard installation, only the drive can be selected. The directory is always `\STEP5\S5_HOME\..`

Search Order

Due to the separation into a system and a home directory, original files (as supplied) and files modified by the user are maintained separately. Files shipped with the package that are modified by the user are stored in the home directory (applies only to Version 7.0 packages!). This means that the original files are retained in the system directory.

Due to this strategy, there is a fixed order when searching.

- STEP 5/ST V7.0: Files are searched for first in the home directory and then in the system directory.
- COM packages: Here, remember that the system directory V7.0 is different from the system directory of the COM adapter. The COM packages use their own system directory ...\S5_SYS\S5_COM.

This division is necessary to allow the COM packages to run.

The printer parameter files (*DR.INI) and path files (*AP.INI) shipped with the package are kept in their own subdirectories below the system directory to maintain a clearer structure.

File:	First	2nd attempt	
`??????DR.INI`	Home directory	`System directory\DR_INI`	for V7.0 packages
`??????AP.INI`	Home directory	`System directory\AP_INI`	for V7.0 packages
`STEP5.S5K`	Home directory	`System directory`	for V7.0 packages
`S5KXS06X.S5K`	Home directory	`System directory\S5_COM`	for COM-packages
`??@@@@??.INI`	Home directory	`System directory`	for V7.0 packages

25.4 STEP 5 Files

Overview

This section is an overview of the directories that contain files relevant to STEP 5. For detailed information about directories and files on your device, refer to the product information.

`C:\STEP5`

Standard setting during installation:

`C:\STEP5\S5_SYS`

STEP 5 system directory with the STEP 5 basic package

`C:\STEP5\S5_HOME`

The S5.BAT file with which you start the STEP5 basic package and P tools.

`C:\STEP5\S5-SYS\EXAMPLE`

This directory contains a sample program with program blocks and an assignment list.

`C:\STEP5\S5_SYS\S5_INST`

Contains installation components, backups of individual S5 program components

`C:\STEP5\S5_SYS\S5_COM\PG_PG`

Link between two PGs for exchanging STEP 5 blocks and files.

`C:\STEP5\S5_SYS\S5_COM`

Default directory for optional packages (COMs), COM DB1,STL, Batch

25.4.1 Functions of Certain STEP 5 Files

Overview

The list below contains 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 filenames stand for characters that can be selected by the user.

Paths	Settings
S5 MEMORY.DAT	Stores the last values entered in job and list boxes.
S5@@@@CF.INI	(STEP 5 Configuration File) This contains the path and the name of the ??????PX.INI file last used. Location: STEP 5 home directory.
??????PX.INI	Data from the project settings.
Programs	
??????ST.S5D	STEP 5 program file.
Assignment list	
??????Z0.SEQ	Sequential assignment list.
??????ZF.SEQ	Assignment error list: list of the errors when translating ??????Z0.SEQ- into the ??????Z0.INI file.
??????Z0.INI	Symbols file, translated assignment list.
??????Z#.INI	Assignment list index files (# = 1 or 2).
??????ZT.SEQ	Stores function key assignments.
Printer	
??????DR.INI	Printer parameters
??????F1.INI	Footer file (80 characters)
??????F2.INI	Footer file (132 characters)
??????LS.INI	Print to file
Specific files	
??????XR.INI	(Reference list) cross reference list
??????SU.INI	Doc commands (submit) for documentation
??????SF.INI	Submit error list
??????TX.INI	Key macros
Bus selection	
??????AP.INI	Path file containing edited bus paths.

25.5 Available Blocks and Parameter Limits

Block		Parameter limits		Comment
Name	STEP 5 name	Input/ output at PG	Call in program	
Organization block	OB	1 - 39	0 - 255	max. 4096 segments per
Program block	PB	0 - 255	0 - 255	block;
Sequence block	SB	0 - 255	0 - 255	**- Length max. 4096 words**
Function block	FB	0 - 255	0 - 255	per block;
Extended function block	FX	0 - 255	0 - 255	– per segment 256 statements (words)
Data block	DB	0 - 255	0 - 255	– max. 2048 DW per block (with header)
Extended data block	DX	0 - 255	0 - 255	– max. (6 • 256)+40 blocks per S5D file
Comment block for OB	OC	1 - 39	–	– Size: max. 16 Kbytes
Comment block for PB	PC	0 - 255	–	
Comment block for SB	SC	0 - 255	–	**– max. (6 • 256)+40**
Comment block for FB	FC	0 - 255	–	**blocks per S5D file**
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	1 - 39	–	
Segment comment for PB	#PBDO	0 - 255	–	**– Size: max. 16 Kbytes**
Segment comment for SB	#SBDO	0 - 255	–	**8 Kwords**
Segment comment for FB	#FBDO	0 - 255	–	**per block**
Segment comment for FX	#FXDO	0 - 255	–	**- max. 255 blocks per S5D file**
Segment comment for DB	#DBDO	0 - 255	–	
Segment comment for DX	#DXDO	0 - 255	–	
Plant comment	#Name	# and max. 8 chars	–	
Variables block	VB	1 - 255	–	PLC function

Max. size S5D file: 4Mbytes
LAD + CSF: max. 400 screen elements per block, max. 50 lines / 8 columns

Appendix A

Chapter Overview

A.1 Key Assignment

Overview

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.

A.1.1 Keys in LAD/CSF

Table A-1 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	
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***.

Table A-1 Function control keys, continued

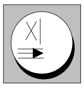
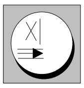
Key name	Key	Output	Edit	Remarks
Insert segment		A segment is inserted before the current segment. An empty screen is displayed and you change to the editing mode	Disabled	In *“Output”* also in *segment functions* with ***SHIFT F4***. In *segment functions* the segment is written to the buffer file.
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.

Table A-2 Terminating Keys

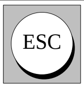
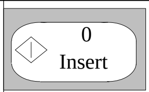
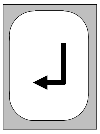
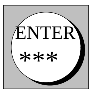
Key name	Key	Output	Edit	Remarks
Cancel (escape)		Changes back to the previous level	Modifications within a field can be cancelled. Otherwise you change to “Output”. Newly entered segments are deleted.	If you exit “Edit” the segment is displayed in its old form. If the segment has been input as a new segment, the previous one is displayed. Also with ***F8***.
Insert		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		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***.

Table A-3   Control Keys

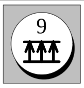
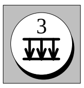
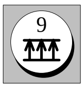
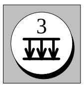
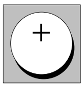
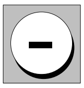
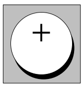
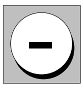
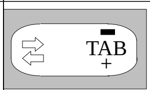
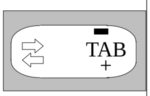
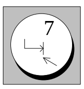
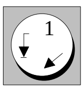
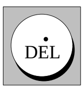
Key name	Key	Output	Edit	Remarks
Page up		Moves the displayed segment one line up.	As “Output”	In list boxes one page up.
Page down		Moves the displayed segment one line down.	As “Output”	In list boxes one page down.
SHIFT Page up	SHIFT 	Moves the displayed segment one page down.	As “Output”	
SHIFT Page down	SHIFT 	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.	In “Output” also in the ”segment functions” with ***F2***.
Segment start	SHIFT 	Disabled	Jump to the start of the displayed segment.	
Input field end		Disabled	Jump to the end of the input field on which the cursor is positioned.	
Input field start	SHIFT 	Disabled	Jump to the start of the input field on which the cursor is positioned.	
Horizontal expand		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*** = ***Exp Hor***
Vertical expand		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		Disabled	Deletes a single character marked by the cursor.	

Table A-3 Control Keys

Key name	Key	Output	Edit	Remarks
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.	
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.	

Table A-4 Special Keys

Key name	Key	Output	Edit	Remarks
Connector = ***F9***	F9	Disabled	Inputs a connector at the current cursor position	Also ***F5*** = *Bin Oper* and ***F4*** *= #*
Negated connector = ***F9***	SHIFT F9	Disabled	Inputs a negated connector at the cursor position	Also ***F5*** = *Bin Oper* and ***F5*** = /.
Not defined *?*	?	Disabled	Input fields are marked as undefined when this key is pressed first after selecting the input field	

A.1.2 Key Assignment STL

Overview

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*.

Table A-5 Key Assignment STL

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***.
Cursor right	⇒ 6 →	Disabled	Move right within an input field. At the end of the field, jump to the first position of the next input field.	
Cursor left	⇐ 4 ←	Disabled	Move left within an input field. At the end of the field, jump to the last position of the previous input field.	
Change to next input field	SHIFT ⇒ 6 →	Disabled	Move to the next input field of the STL line.	
Change to previous input field	SHIFT ⇐ 4 ←	Disabled	Move to the previous input field of the STL line.	

A.2 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 list 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 list boxes with ***F3*** = *Select*.

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). Settings include the following:

- storage location for the various files
- method of representation (LAD/CSF/STL)
- files required for the project
- mode
- parameters for printing out, etc.

Settings

In the pages of the project settings, you enter the files and parameters for your project. This box is divided into six pages.

Tabs

The selected parameters and files are later entered in the corresponding job and list boxes. The files and parameters selected on these pages are valid for all the work in the entire project.

In the tab pages, you can position the ***cursor*** using the cursor keys or the mouse. By double clicking on the parameters, you can either open a list box or change the default. You can also make selections by pressing the ***F3*** key twice.

Menu Commands

File

Project >	
Set	Before you start the actual programming, you set all the parameters required for a project in the tab pages of the project settings.
Load ...	A project file created with the set function is loaded. Loading the file makes all the settings it contains valid for your work. The previously valid settings are overwritten.
Save	This saves all the settings made in the *project settings* tab pages in the current project file (*PJ.INI).
Save As ...	Save the settings in a new (selectable) project file (*PJ.INI).
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 ...	Deletes blocks on the PG and PLC, documentation files only on the PG. Triggers a PLC overall reset
Compress	STEP 5 blocks in the program file are checked and compressed.
DOS file >	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 file >	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 PCP/M ->DOS ...	This converts PCP/M files to S5-DOS ST/MT files.
Copy DOS -> PCP/M ...	This converts STEP 5 files created with S5-DOS ST/MT into PCP/M files.
Delete ...	PCP/M files on a PCP/M medium are deleted.
DOS Commands	With this function, you change to the DOS level
Exit	You exit STEP 5/ST.

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 Block ...	With this function you supply parameters and start the editor for data blocks.
DB Screen ...	With this function you supply parameters and start the editor for DB screens.
Assignment List	As soon as you activate this function, the editor for the assignment list is called directly.
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 Bus Path* job box.
Printer Parameters	You create a control character record for your particular printer which is stored in a printer file.
Footer Editor	With this function you can create a new footer file or modify an existing file.

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 *project settings* pages in 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* list 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.
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.
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* list 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.

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 PLC Memory	With this function you can eliminate invalid blocks on the PLC and shift the valid blocks together.
PLC Info 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.
PLC Info 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 PLC Memory	This function outputs the absolute addresses of the PLC and their contents on a selectable medium.
PLC Memory Conf	This outputs the memory configuration and indicates how much of the user memory in the PLC is currently occupied.
PLC Sys Parameters	This displays the system parameters of the PLC on the screen.

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 settings for the individual functions must already be made in the *project settings* tab pages.

Make 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.
EPROM Handling	With this function you can transfer (blow) STEP 5 programs from the selected program file to EPROM/EEPROM submodules. The *EPROM Programming* box is displayed.
Automatic Rewiring ...	The operands are renamed automatically based on a modified or new assignment list. The *Automatic rewiring* job box 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 Rewiring ...	You rename operands in an operand list. The *Manual rewiring* job box is displayed. Here, you select the new program file name *to program file*. Following this, you enter the operands in a table.

Assignment Lists > 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 assignment list to the corresponding symbols file. You enter the name of the source file to be converted in the *Convert assignment list SEQ > INI* job box.

Convert INI >SEQ — You convert the symbols file to the corresponding assignment list, 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 symbols file 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 symbols file* job box. Following this, you can correct the symbols file.

Convert V1.x and V2.x — Symbols files created with earlier versions (V1.0, V2.0) can be converted.

Delete SEQ — This function deletes an assignment list.

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.

Convert ... With this function, project data are converted from the file format of STEP 5/ST Version 6.x to the format of version 7.x.

Language ... You select the language you want to work in.

Colors ... You can change the colors of the screen displays.

Documentation

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 block(s)* job box.

Data Blocks ... You can output either individual or all the data blocks of a program.

DB Screens ... This function outputs data blocks containing screens. Select the blocks in the Output *DB screens* job box.

Assignment List ... You output an assignment list. If the assignment list is not already set, you can select it in the *Print assignment list* job box.

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.

Cross References ... 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.
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.
DB1 Screens ...	With this function, you can output data blocks that contain screens to a printer or file.
Block List ...	This function prints out a list of all the program and data blocks of the set program file.
Assignment List ...	You output an assignment list. You can print this either in sequential form as edited or sorted according to absolute/symbolic operands.
Program Structure	This prints out the call structure of the individual blocks in a program file.
Cross References ...	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.
S Flag List ...	With this function, you output the I/Q/F list of S flags.
Checklist ...	This function checks through the configuration data. Depending on the option, the free operands, operands without symbols, and operands without setpoint data for the I/Q/F operands.
Text File ...	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 submit file.
Test ...	Doc commands in a selected file are checked to make sure they can be executed. Errors are indicated and saved in an error file.
Output Log File ...	With this function, yo can output the log file created during the test.
Run	The doc commands in a file are started.
Output ...	You can print the content of a doc command file.
Edit Structure	This provides you with information about the links between individual doc command files. At the same time, you can edit individual DOC command files.
Output Structure ...	The structure of linked doc command files is output in A4 or A3 format on a printer or to a file (LS.INI).

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.
COM DB1	You change to the COM DB1 parameter assignment software. With this package, you can assign parameters to CPUs of the lower and mid range of performance, while being sure that no parameter errors occur.
AWL Batch	You change to the STL editor/batch compiler package.
Others ...	You select the S5 package you want to activate in the *Other SIMATIC S5 Programs* list box.

Help	With these functions, you can display the following information:
Key Assignment List ...	This displays information about the function keys These are keys with which you can activate certain functions directly.
About STEP 5/ST 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.
User Interface >	This menu command provides you with descriptions of ways in which you can obtain information about certain topics.
Using Menus	Help and information about using menus.
Using Dialog Boxes	Help and information about using dialog boxes.
Using Project Settings	Help and information about the tab pages of the project settings.
General Information	General information about working with the user interface.
Compatibility	Information about the compatibility of STEP 5/ST V 7.0 with earlier versions.
Notes	Notes on special topics.

A.3 Key Macro

Overview

Using the key macro program, you can record key sequences in the block editor. The sequences are saved in the S5 file ??????TX.INI. This file is on the drive in which STEP 5 was installed. The name ?????? can be freely selected by the user.

Selection

You select a key macro file to record or play a macro using dialog boxes in the dialog language selected for STEP 5.

Playing

A key macro can be run step–by–step. In the dialog box, it is also possible to assign a macro title and a comment. You can edit these at any time. File names and macro titles are displayed in a dialog box allowing fast simple selection. You can save key macros in any directory.

Macro Function

When the macro function is active (recording or playing a key macro) the current mode is displayed in the right–hand top corner in English.

The following displays are possible:

`REQU`	Request	Request macro mode
`RECI`	Record Init	Initialize recording
`REC`	Record	Record
`RECA`	Record Abort	Abort the recording
`RECE`	Record End	Terminate the recording
`PLAI`	Play Init	Initialize play
`PLAY`	Play	Play the macro
`PLAA`	Play Abort	Abort playing the macro
`PLAE`	Play End	End of playing the macro

Startup Macro

The macro with the name START@TX.INI is a special case. This startup macro is started automatically when you call STEP 5/ST providing the key macro file START@TX.INI is located in the home directory. If necessary, you can create the startup macro yourself.

Special Case

Downwards compatibility to the Version 6.6 key macros is not possible due to the changed user interface of STEP 5/ST and the new recording format.

The hotkeys (CTRL– A, CTRL– E, CTRL– D), as used in Version 6.6 are no longer used for the key macro functions (record, play).

When using the hotkeys, you should note that the key assignment varies from language to language.

It is not possible to use the mouse to operate STEP5/SR during recording.

The following new hotkeys are now available with Version 7.0:

Table A-6 Operations

Key Macro	Function
`CRTL+ALT+D`	When used in the ”Normal Mode” (no mode display),this calls the ”Select Macro” dialog box.You can select a macro for recording or playing.
`CRTL+ALT+D`	During the recording of a macro (mode display ”REC”)this stops the recording.
`ESC`	During the playing of a macro (mode display ”PLAY”)this stops the play mode and aborts the currently activekey macro.
`CRTL+ALT+T`	If you selected the single step playing of a macro in the ”Select Macro”dialog box, you can play the macro step–by–step (in other words key–by–key) using this hot-key. Each step in the macro must be confirmed by the key combination CTRL+ALT+T This function is extremely useful when checking that a macro does what it is inten-ded to do. The single step mode is not displayed separately.

Recommendations for the Use of Key Macros

The key macro function is intended mainly for keyboard sequences that you use regularly in the editors. Automated sequences within menus and dialog boxes including changes to packages can only run correctly when the conditions at the time of playing the macro are the same as the conditions when you recorded the macro. For this reason, it is advisable to restrict the use of key macros to limited tasks where the conditions can be checked easily.

Note the following points when using key macros:

- Central start point:

  Create a few start points within the packages where you start or play your key macros and document these points in the key macro comment.

  Examples of typical start points:

  Within the menus FILE menu item not open
  Within editors Correction mode

- Fast selection:

  Select a macro title to indicate the purpose of the key macro.This allows a faster selection in the ”Macro Selection” dialog box.

- Correct start point and necessary conditions:

  Document the start point or required conditions (for example STL) for playing the key macro in the macro comment.

Recording Key Strokes Within the User Interface.

Menus You should only use the following keyboard input within the user interface:

`ALT+<letter>` for changing to the corresponding menu

`<letter>` for selecting a menu item in the selected menu

Acceleration keys (function keys combined with UNSHIFT, SHIFT, CTRL and ALT) for a direct jump to the most important menu items.

Do not under any circumstances use display control keys (cursor keys, tab stop etc.) before operations within the user interface.

Dialog boxes

Under no circumstances use the display control keys (cursor keys, tab stop etc.) for operation within dialog boxes.

Do not use check boxes during the recording.

Prior to recording the macro, make all the selections necessary in check boxes in the dialogs.

Checking key macros:

After creating a key macro, you can check its effects by playing it in the single step mode, key–by–key.

Note

The recording or playing of a macro is interrupted when other packages are selected with the menu items ”Change/Others...”, ”STL Batch”, ”COM DB1” and ”DOS Commands” and is resumed after you return from the packages.

Keyboard Editor

In STEP 5/ST, you can reassign the functions of the F keys using a keyboard editor. When you save the resulting S5K files, remember the search order.

The keyboard files STEP5.S5K and S5KXS06X.S5K are shipped with STEP 5 in the system directory\S5_INST. After copying these two files to the home directory, you can adapt them using the keyboard editor.

When you call STEP 5/ST, the keyboard file STEP5.S5K is used for the STEP 5/ST sections and the keyboard file S5KXS06X.S5K for the STEP 5/ST V6.x sections (COM adapter).

The functions of most F keys can be reassigned with the keyboard editor.

Note

Please note that the key combinations ALT–<number> and ALT–<letter> must not be assigned a function because these keys are already required by the STEP 5 package. This also applies to the key for the macro function.

A.4 Programming Rules

Overview

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.

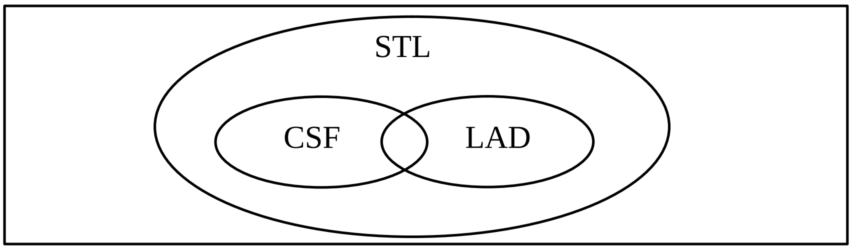


Figure A-1 Scope and Limits of the STEP 5 Methods of Representation

---

Note

Programs you have written in LAD or CSF can always be translated back to STL.

A.4.1 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.

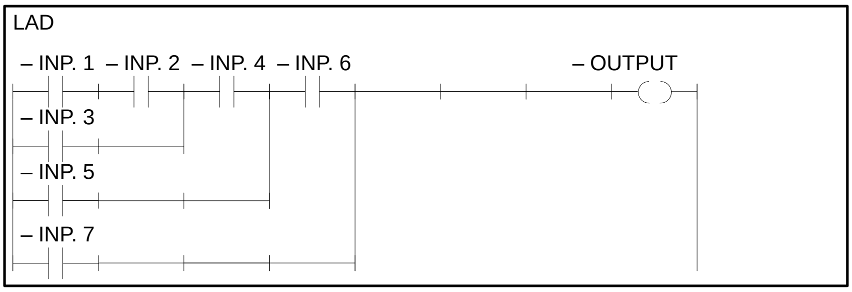


Figure A-2 Example of Nesting when Inputting in LAD

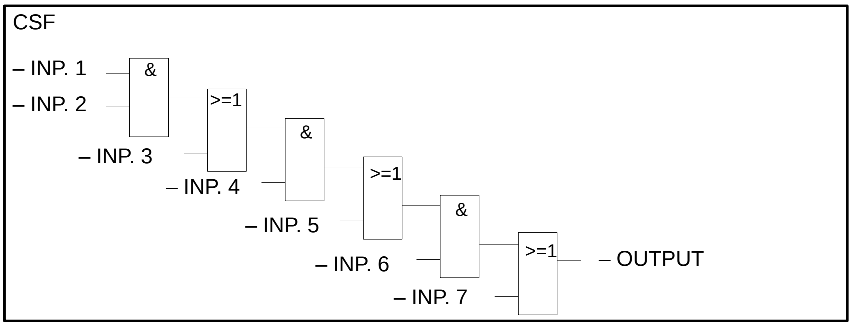


Figure A-3 Example of the Nesting above Output in CSF

Input in CSF, Output in LAD

Too many entries in a CSF box can exceed the display limits (8 levels) in LAD.

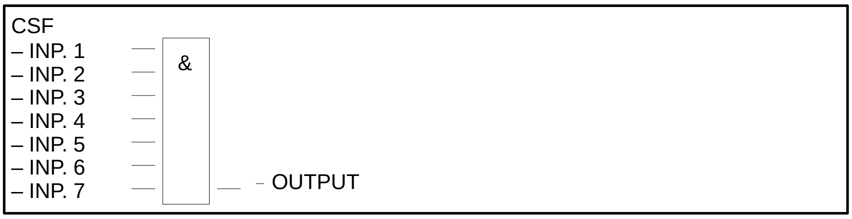


Figure A-4 Example of Nesting when Inputting in CSF

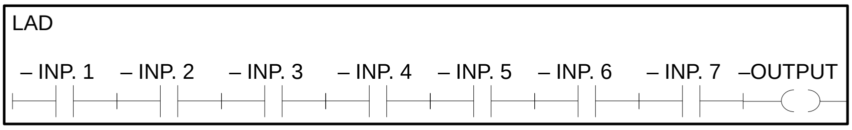


Figure A-5 Example of the Nesting above Output in LAD

Output of a Complex Element

The output of a complex element (latch, comparator, timer or counter) must not be ORed.

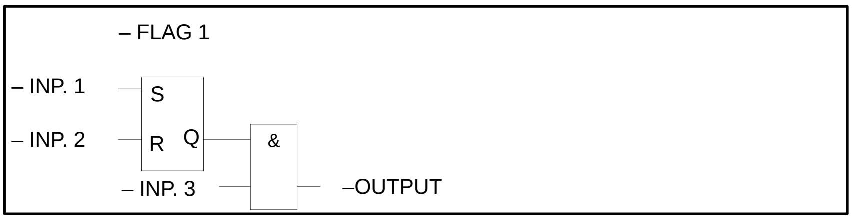


Figure A-6 Only AND Boxes are Permitted after a Complex Element

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

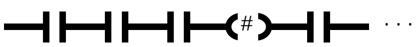
LAD	STL
F . . .	: A F . . .
—(#)—	: = F . . .

Figure A-7 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

LAD



STL

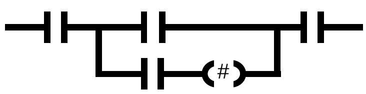
```
: A
: A (
: A
: =  F
: A   F
: A
  :
```

Figure A-8 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-9 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-10 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

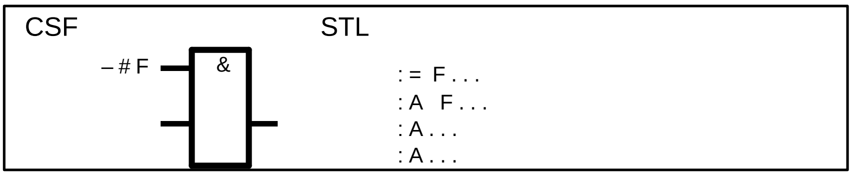


Figure A-11 Connector at the First Input

The connector is not within parenthesis.

Connector not at the first input of an OR box

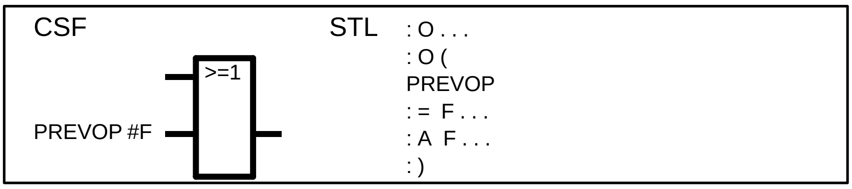


Figure A-12 Connector not at the First Input

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

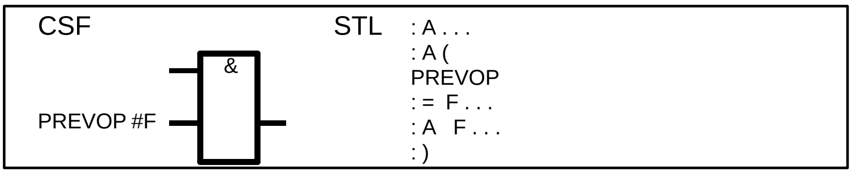


Figure A-13 Connector not at the First Input

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)

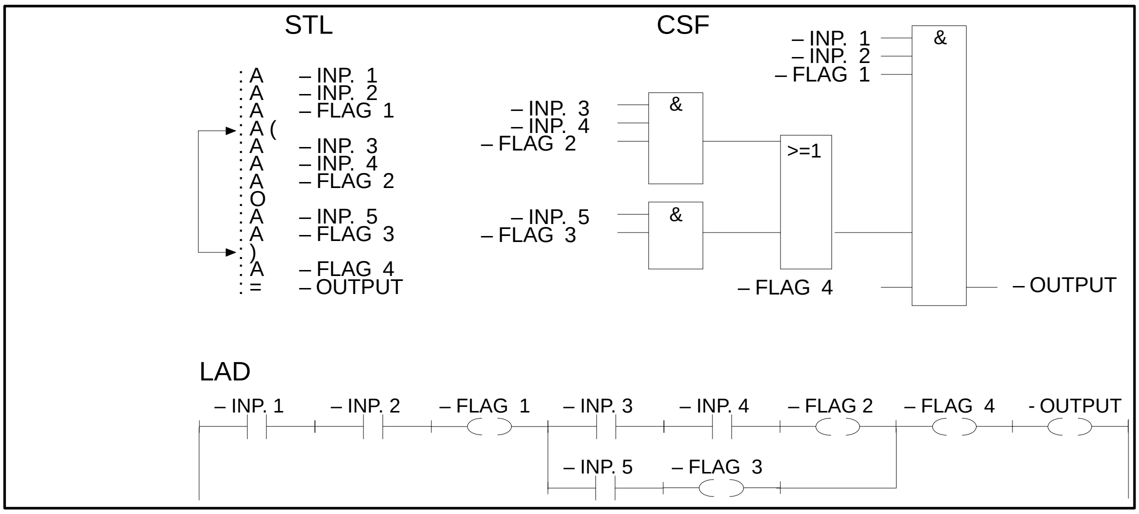


Figure A-14 Example without Connectors

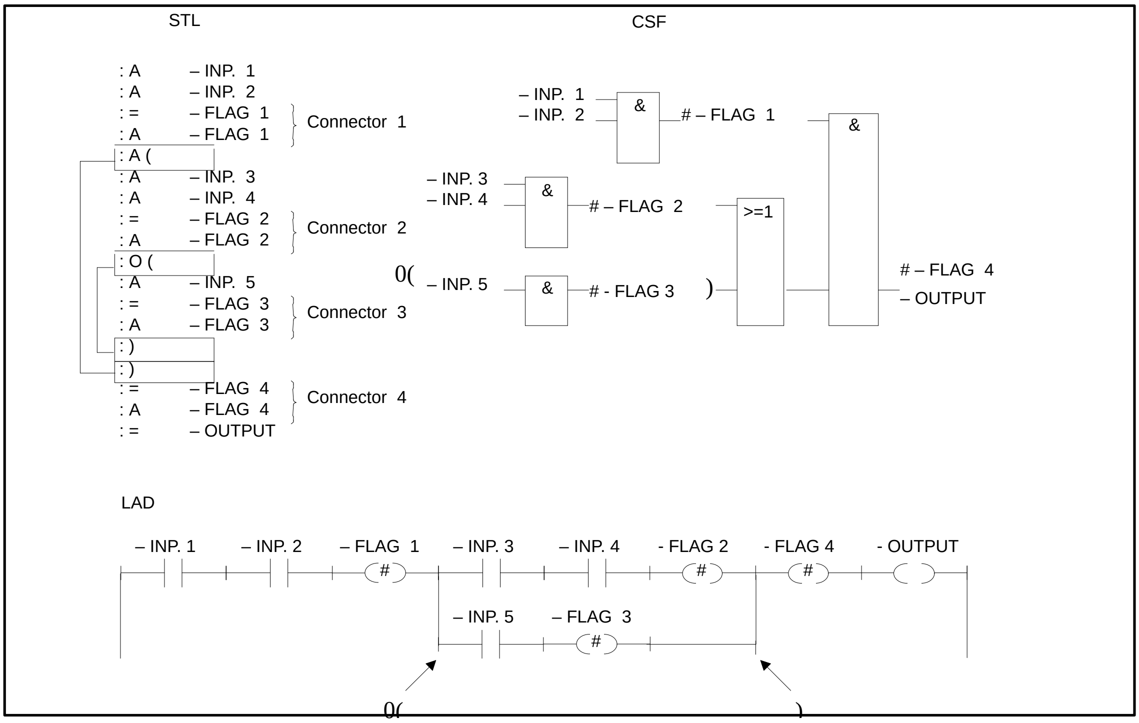


Figure A-15 Example with Connectors

A.4.2 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.

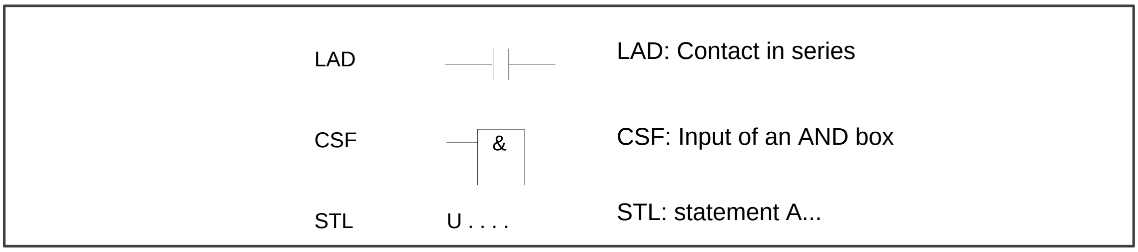


Figure A-16 AND Operation

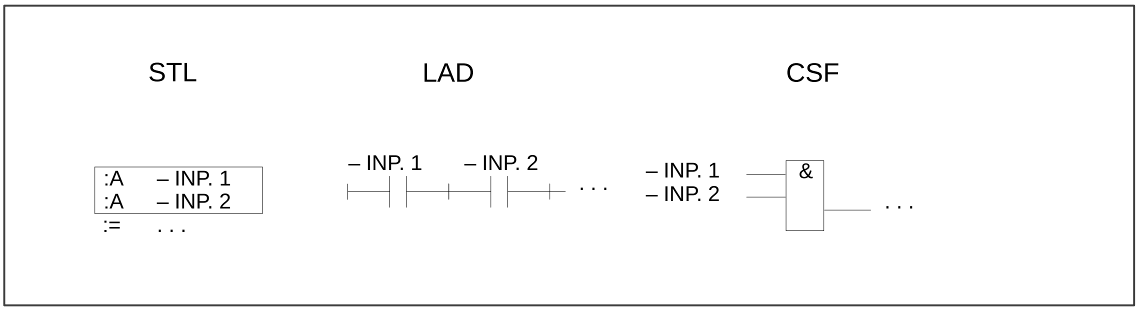


Figure A-17 UND Operations in STL, LAD, CSF

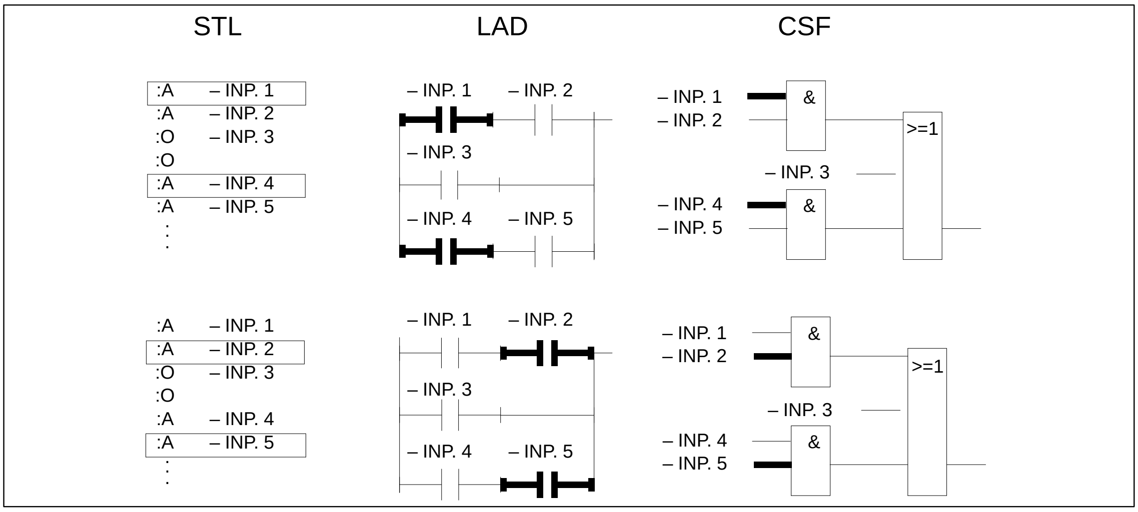


Figure A-18 Example of the Rule for AND Operations

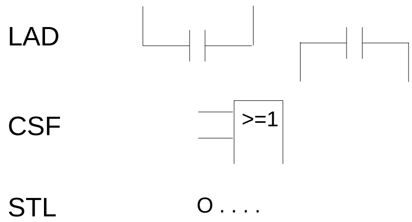
OR Operation

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...



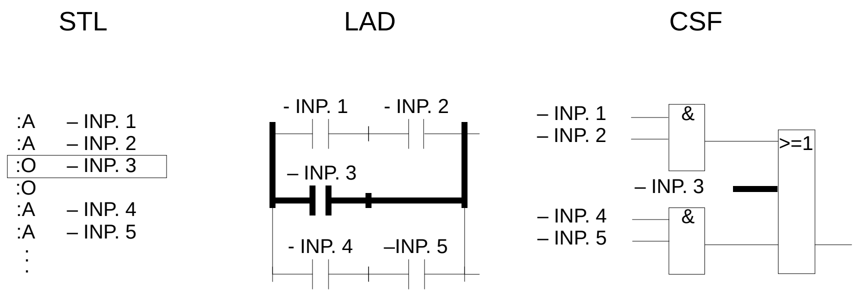
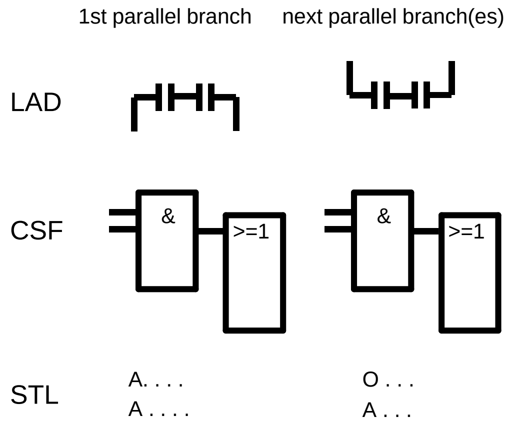


Figure A-19 Example of the Rule for OR Operations

AND before OR Operation



LAD:	more contacts in a parallel branch	
CSF:	AND box before OR box	
STL:	statements	O ...
	parallel branch	A ...
		A ...

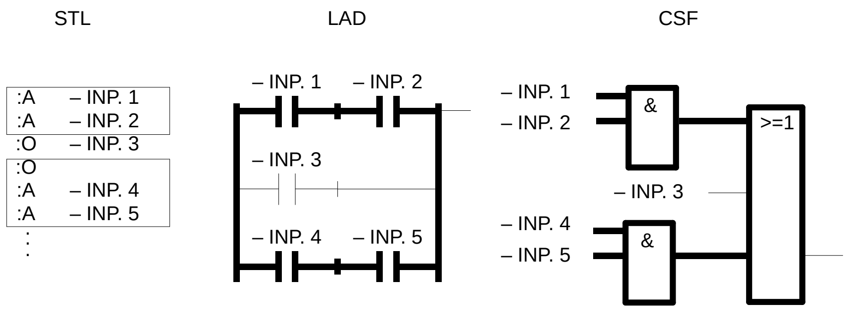
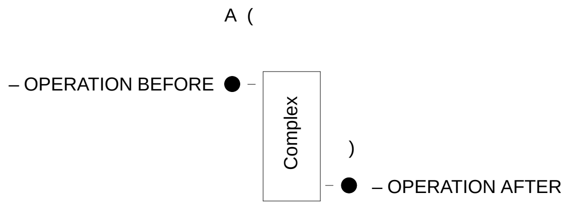


Figure A-20 Example of the Rule for an AND before OR operation

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

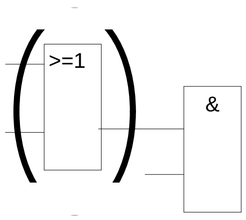
STL

```
A(
O . . .
O . . . .
O . . .
)
A . . .
```

STL: statements	A(
	OR operation
	)
	A

LAD: Connect parallel contacts in series.

CSF



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)

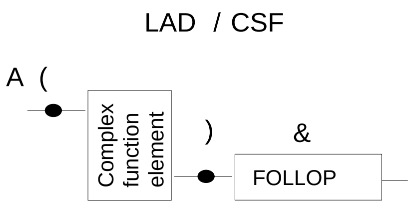
The following rules apply to complex elements:

– no following operation: no parenthesis

– AND operation follows: A (...).

– OR operation follows: O (...), only for CSF.

– Complex elements cannot be followed by other operations.



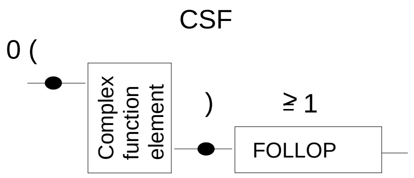


Figure A-21 Parenthesis with Complex Elements

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.

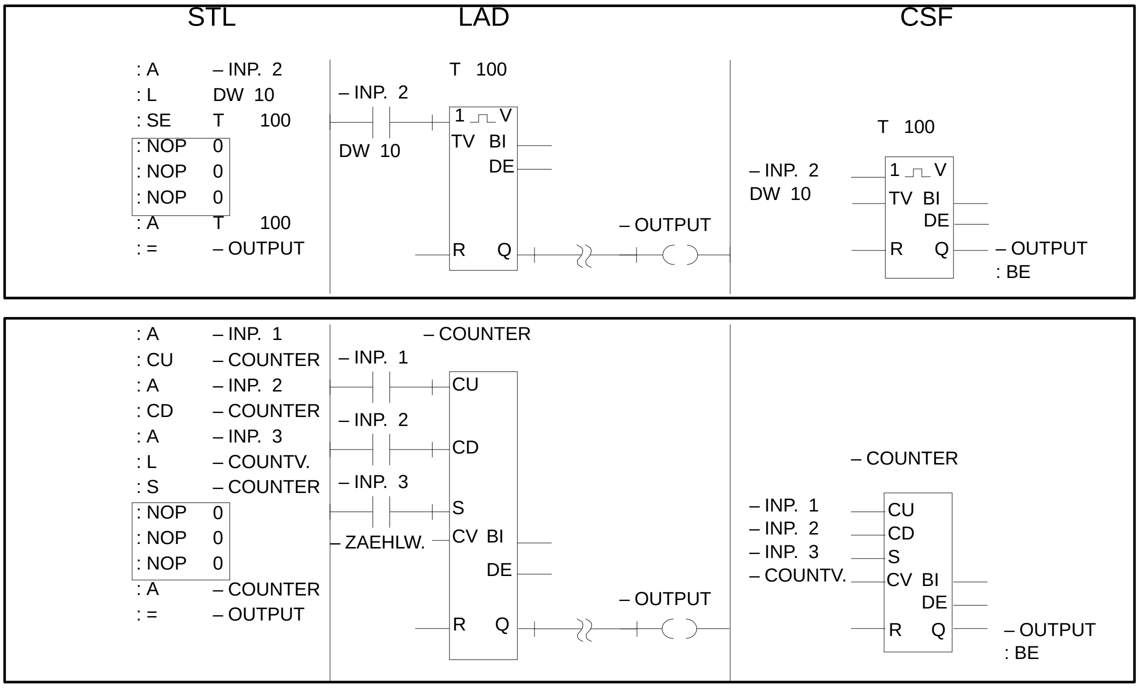


Figure A-22 Example of Undefined Inputs and Outputs in STL, LAD and CSF

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 protection

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 project settings.

Actual operand

The actual operand (parameter list in the calling block) replaces 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 (*Editor, 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 poin-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 list box and then return to STEP 5 at any time.

Comment 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, Make 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.

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, of program file” or ”of PLC”, the block list of a program file is displayed or printed out. The block type, number, length and the library number (not if PLC is selected) of each block is displayed.

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).

Documentation file (DOCFILE)
The documentation file (#NAME) contains block-specific plant comment.

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. During a session with an editor, other editors are also available.

EPROM handling
This is a utility that can be started in the ”Management” menu and is 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 labeling 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 *project settings* (Documentation page).

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 (keys ***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 list 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 Five digit number to identify blocks (block number)

List 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.

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, the language option and the submenu for screen color settings.

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 displays 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 labeled.

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 PCPM 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 PCPM 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 *project settings*. Its extension is *DR.INI and it is stored in the system directory.

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.

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 block → Block

Programming number

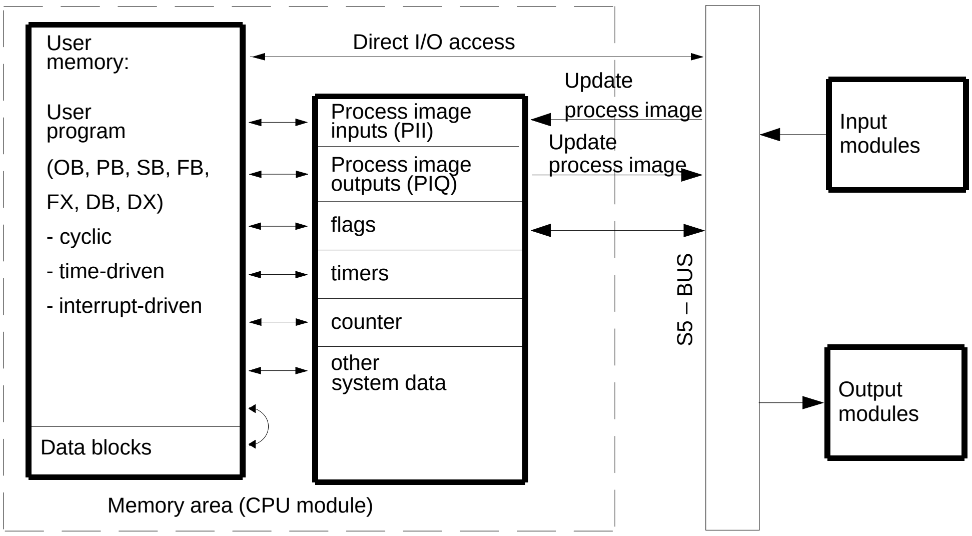
This is used to identify the type of EPROM/EEPROM plugged in. This is assigned to the order number of the specific submodule. When a function is invoked (e.g. blow EPROM), STEP 5 examines the programming number and then displays the parameters of the submodule. This avoids errors when submodules are 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 (*PX.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



Project settings

Settings tab pages selected in the *File, Project* menu to define a project by 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.

R

Result of logic operation RLO

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, Automatic/Manual Rewiring” 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 of the segment. This identifier must not be modified or deleted otherwise the assignment of comment to block is lost.

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 to it 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 resources. 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.

Standard function block

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 DB0.

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). This status function is an online function and is selected in the ”Test” menu.

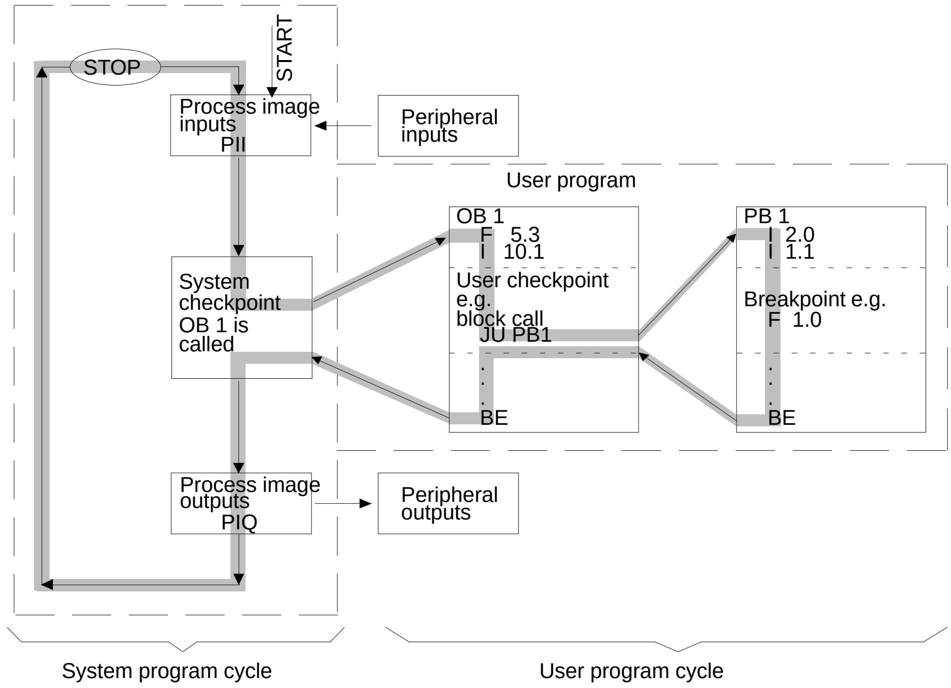
Symbols file

The list of the assignment of symbolic to absolute operands stored in a source file. Blocks programmed using symbolic operands are converted to absolute 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 SYSID

The SYSID file (Settings, Page 1) contains identification data and characteristics, e.g. for the communications processors (CPs).

T

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.

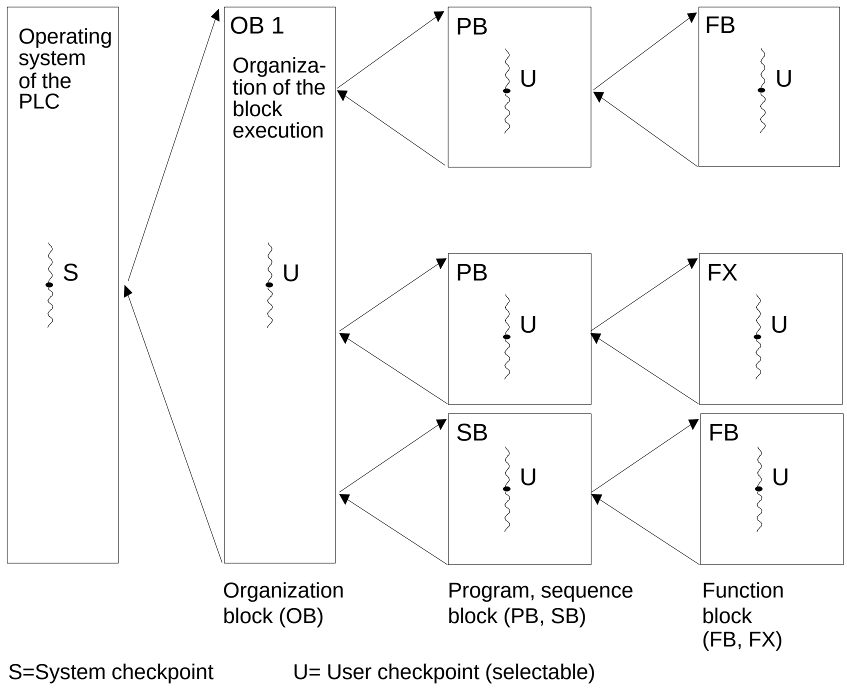
Text editor

Tool for creating and working with segments and operand comments in Documentation blocks. Documentation blocks are called using the job/list box in the STEP 5 block and data block editor.

U

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 variable* or *program test ON*).



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.

W

Wildcards

* = Placeholder for a character string or format-dependent name.
? = Placeholder for a character.

Index

D

E

F

G

H

I

M

N

O

P

T

U

V

W

X

Siemens AG
AUT E 146

Östliche Rheinbrückenstr. 50
D–76181 Karlsruhe
Federal Republic of Germany

From:
Your Name: _
Your Title: _
Company Name: _
Street: _
City, Zip Code _
Country: _
Phone: _

Please check any industry that applies to you:

❒ Automotive
❒ Chemical
❒ Electrical Machinery
❒ Food
❒ Instrument and Control
❒ Nonelectrical Machinery
❒ Petrochemical
❒ Pharmaceutical
❒ Plastic
❒ Pulp and Paper
❒ Textiles
❒ Transportation
❒ Other _ _ _ _ _ _ _ _ _ _ _

Remarks Form

Your comments and recommendations will help us to improve the quality and usefulness of our publications. Please take the first available opportunity to fill out this questionnaire and return it to Siemens.

Please give each of the following questions your own personal mark within the range from 1 (very good) to 5 (poor).

1. Do the contents meet your requirements? ☐
2. Is the information you need easy to find? ☐
3. Is the text easy to understand? ☐
4. Does the level of technical detail meet your requirements? ☐
5. Please rate the quality of the graphics/tables: ☐

Additional comments: