

# Festo Software Tools

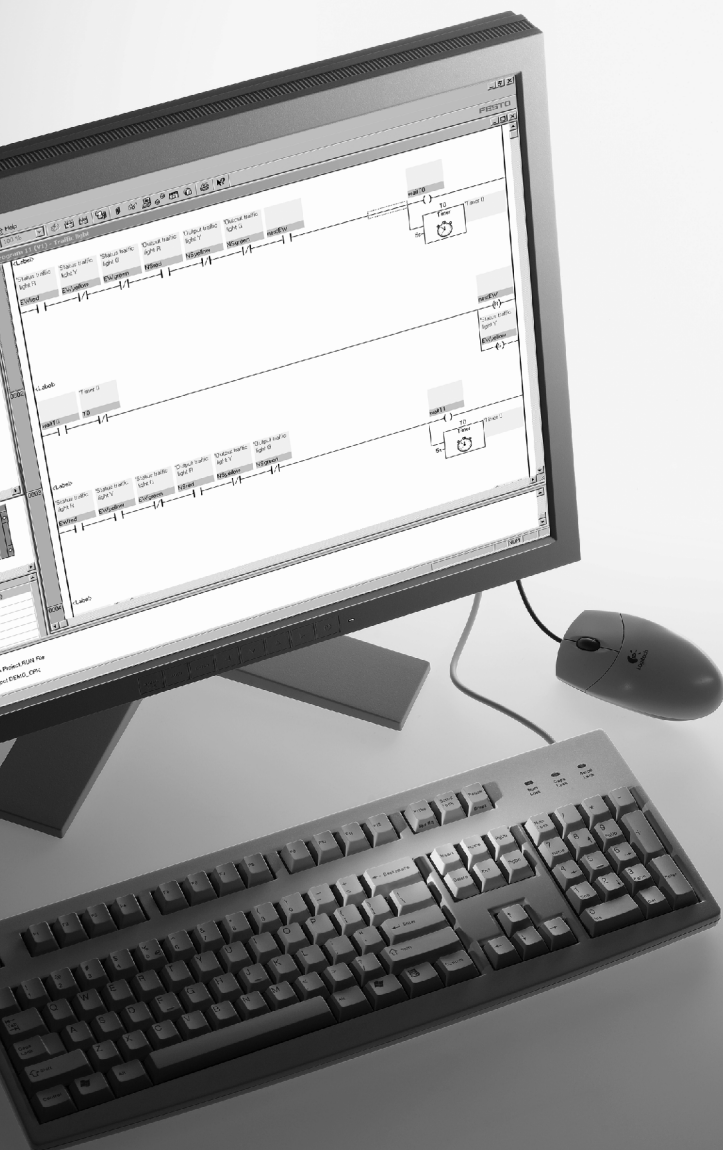
# FESTO

**Software package  
FST**

Version 4

Volume 1

Programming in  
Statement List and  
Ladder Diagram



**Manual**  
en 0403NH  
[682 297]



## Contents and general instructions

Authors ..... S. Breuer, I. Walter, O. Westrik

Editor ..... M. Holder

Original ..... de

Edition ..... en 0403NH

Designation ..... P.BE-FST4-B1-DE

Software package Order no. .... 682 297

© (Festo AG & Co. KG, D-73726 Esslingen, 2004)

Internet: <http://www.festo.com>

E-Mail: [service\\_international@festo.com](mailto:service_international@festo.com)

The copying, distribution and utilisation of this document as well as the communication of its contents to others without expressed authorisation is prohibited. Offenders will be held liable for the payment of damages. All rights reserved, in particular the right to carry out patent, utility model or ornamental design registrations.

Microsoft® Windows® is a registered trademark of the  
Microsoft Corporation

Microsoft Internet Explorer® is a registered trademark of the  
Microsoft Corporation

Pentium is a registered trademark of the  
Intel Corporation

## **Festo terms and conditions for the use of software packages**

### **I. Property rights and scope of use**

The product comprises data processing programs and the corresponding program descriptions. It is described below in its entirety as the software package.

Festo or third parties own property rights to these software packages. In as far as these rights extend to third parties, Festo has appropriate rights of use. Festo permits the purchaser use subject to the following conditions:

#### **1. Scope of use**

- a) The purchaser of the FST software package is entitled to use it for his own, internal purposes as a tool for controlling Festo products on any number of machines within his business premises (place of use). This right of use comprises exclusively the right to run the software package on the central processors (machines) employed at the place of use.
- b) The conditions under a) will also apply insofar as the software package is combined with other programs or run in combination with same.
- c) The software package, even if used in combination with other programs, may be copied and saved any number of times for data backup and trouble shooting purposes at the purchaser's place of use. Copying for any other purpose, specifically for disclosure to third parties not covered by the regulation in Item 3, as well as any processing or use other than those permitted above is illegal.
- d) Any further use, specifically copying for other purposes and disclosure to third parties not covered by the regulation in 3 and also any processing or other use, is illegal.

#### **2. Copyright note**

Every program contains a copyright note. This note must be transferred to every copy, edit and every part of the program that is combined with other programs.

### **3. Transferring the authority of use**

The purchaser can transfer his authority of use in its entirety to a third party to the extent and under the limitations of the conditions in accordance with items 1 and 2. The third party must be made explicitly aware of these conditions.

Upon transfer, all rights of use of the transferor lapse, specifically rights to the copies, edits and combined programs. Insofar these are not transferred to the third party, they shall be destroyed.

### **4. Any conditions of other manufacturers contained in this software package are invalid.**

## **II. Exporting the software package**

When exporting the software package, the licensee will observe the export regulations of the Federal Republic of Germany and those of the purchasing country.

### III. Warranty

#### **License Agreement**

1. Festo guarantees that the software program it has created is consistent with the description of the application and the program specification. It does not, however, guarantee that the functions contained in the software shall run continuously and trouble free or that such functions can be run in all combinations selected by and under all conditions of use proposed by the licensee and/or satisfy the corresponding requirements.
2. Faults in the software material indicated by the licensee in a transparent, written form within the warranty period shall be rectified by Festo subject to the exclusion of all other warranty claims within an appropriate period by way of supplying a revised version.
3. Should Festo fail to meet its obligation to rectify faults within the appropriate period or should rectification work ultimately fail, the licensee shall be entitled to demand a fair reduction in the utilisation fee or to withdraw from the agreement.
4. The period of limitation for warranty claims is 3 months. The period of limitation commences on transmission/transfer of the license material.
5. The warranty is not valid for defects caused by changes made by the licensee himself to the conditions of use created for the program and indicated in the documentation/specification. If the defect cannot be revealed during an examination or if a defect results from circumstances beyond Festo's control, the licensee shall be responsible for costs incurred by Festo.

6. We stress that current technology does not enable the development of software that runs trouble free and is compatible with all applications and combinations proposed by the user.

The software will therefore usually function as described in the program description and the operator's manual. The software must be designed such that at the time of its transfer or provision, it will function under normal conditions of operation and use.

7. Festo does not guarantee that the software will satisfy all applications and purposes intended by the user/purchaser, specifically that it will run trouble free and is compatible with all other used programs.

The responsibility for correct selection and the consequences of using the software in the environment selected by the user and also the thus intended and achieved results therefore lies with the user/purchaser himself. The same applies to the written material accompanying the software. Use of the programmed software does not, therefore, exempt you as the customer from the obligations and responsibility to observe and comply with computer and security-related conditions and also a comprehensive function test.

#### IV. Liability / Limitations of liability

1. The possibility of damage claims on the part of the licensee and specifically a liability for consequential damage are ruled out, irrespective of the legal justification; this also applies to all claims arising from impossibility, non-fulfilment, positive breach of contract, illegal actions and default.

2. Furthermore, Festo is not liable for inadequate economic success or for third party damage or claims, with the exception of claims arising from the infringement of third party property rights.

3. The limitations of liability as per paragraphs 1 and 2 do not apply if, in cases of intent or gross negligence or lack of warranted characteristics, a compulsory liability exists. In such a case, Festo's liability is limited to the damage discernable by Festo when the definitive circumstances are made known.



## V. Safety Guidelines/Documentation

Warranty and liability claims in conformity with the aforementioned regulations (items III. and IV) may be raised only if the user has observed the safety guidelines of the documentation in conjunction with the use of the machine and its safety guidelines. The user himself is responsible for ensuring that our software package is compatible with the machine employed.



## Contents

Festo terms and conditions for the use of software packages .....	III
Designated use .....	XV
Safety regulations .....	XV
Intended readership .....	XVII
Service .....	XVII
Notes on the use of this manual .....	XVII
Important user instructions .....	XVIII
Current information on FST 4.1 .....	XX
Contents of the software package: .....	XXI
Requirements for the PC .....	XXI
Conventions .....	XXII
Product-specific terms and abbreviations .....	XXIII
 <b>1. Installation and general operator instructions .....</b>	<b>1-1</b>
1.1 FST software package .....	1-4
1.1.1 Installing FST .....	1-4
1.1.2 Deinstalling FST .....	1-6
1.2 Starting FST .....	1-6
1.3 What's new .....	1-7
1.4 The FST operating interface .....	1-8
1.4.1 FST features .....	1-10
1.4.2 Window within the FST .....	1-14
1.4.3 Program editors .....	1-16
1.4.4 Multiple-column list fields in dialog windows .....	1-18
1.4.5 The FST software Help system .....	1-19
1.4.6 Exiting FST .....	1-21
 <b>2. Basic functions of the FST software .....</b>	<b>2-1</b>
2.1 Organising projects .....	2-5
2.1.1 The project directory .....	2-6
2.1.2 General project information .....	2-9
2.2 Working with projects .....	2-12

2.2.1	Creating a new project .....	2-12
2.2.2	Changing the project properties .....	2-15
2.2.3	Opening a project .....	2-16
2.2.4	Closing the current project .....	2-17
2.2.5	Managing projects (delete, copy, rename...) .....	2-18
2.2.6	Preparing the project for importing .....	2-20
2.2.7	Downloading projects into the controller .....	2-21
2.2.8	Updating projects in the controller .....	2-22
2.2.9	Uploading project sources from the controller .....	2-23
2.2.10	Displaying the project file content .....	2-24
2.2.11	Deleting the restorable files .....	2-25
2.2.12	The project documentation .....	2-25
2.2.13	Printing a project .....	2-26
2.2.14	Archiving or restoring projects .....	2-30
2.3	Replicating the hardware configuration .....	2-31
2.3.1	Hardware configuration for PS1, FEC Standard and FEC Compact .	2-32
2.3.2	Hardware configuration for CPX terminal (CPX-FEC) .....	2-36
2.3.3	Online and offline mode of the hardware configurator (CPX-FEC) .	2-38
2.3.4	Save the actual configuration as the nominal configuration (CPX-FEX) .....	2-39
2.3.5	Performing nominal/actual comparison (CPX-FEC) .....	2-41
2.3.6	Manually generating the nominal configuration (CPX-FEC) .....	2-43
2.3.7	Setting and changing the address mapping (CPX-FEC) .....	2-48
2.3.8	Parameterising the CPX terminal with FST (CPX-FEC) .....	2-48
2.3.9	Trace memory .....	2-54
2.4	The driver configuration .....	2-55
2.4.1	Editing functions for driver configuration .....	2-56
2.5	Controller settings (PLC settings) .....	2-60
2.5.1	Settings for runtime behaviour .....	2-60
2.5.2	Drives .....	2-62
2.5.3	Options .....	2-64
2.5.4	Password protection .....	2-65
2.5.5	Settings for downloading .....	2-67
2.6	The allocation list .....	2-69
2.6.1	Functions of the allocation list editor .....	2-72

2.7	The Strings Editor .....	2-75
2.7.1	Functions of the Strings Editor .....	2-76
2.8	Managing the controller programs .....	2-78
2.8.1	Creating a new program or module .....	2-79
2.8.2	Importing programs and modules .....	2-81
2.8.3	Exporting programs and modules into the library .....	2-84
2.8.4	Viewing and changing program and module properties .....	2-85
2.8.5	Deleting programs or modules .....	2-86
2.8.6	Opening a program or module .....	2-88
2.8.7	Compiling programs and modules .....	2-89
2.8.8	Selecting an object for compilation and downloading .....	2-90
2.9	Online mode .....	2-91
2.9.1	Configuring the online link .....	2-92
2.9.2	The online control panel .....	2-95
2.9.3	Setting password through the online control panel .....	2-98
2.9.4	The online display .....	2-99
2.9.5	FST file transfer .....	2-109
2.9.6	The command interpreter terminal .....	2-113
2.9.7	Online mode of the hardware configurator (FEC-CPX only) .....	2-114
2.9.8	STL and LDR Online display .....	2-114
2.9.9	The Web Browser .....	2-115
2.10	Runtime library (drivers and I/O scripts) .....	2-117
2.10.1	Managing I/O scripts in the runtime library .....	2-119
2.10.2	Managing drivers in the runtime library .....	2-121
2.11	External Tools .....	2-124
2.11.1	Inserting or changing a program call .....	2-125
<b>3.</b>	<b>Programming in statement list .....</b>	<b>3-1</b>
3.1	Fundamental principles for programming in the statement list .....	3-4
3.2	The statement list editor .....	3-5
3.2.1	STL editor preferences .....	3-6
3.2.2	Functions of the STL editor .....	3-7
3.3	Structure of the STL programs .....	3-20
3.3.1	Step program .....	3-21

3.3.2	Parallel logic program .....	3-23
3.3.3	Executive part .....	3-24
3.4	Brief description of the STL instructions .....	3-25
3.5	STL online display .....	3-39
3.5.1	Functions of the STL online display .....	3-40
3.5.2	Deactivating STL online display .....	3-46
<b>4.</b>	<b>Programming in ladder diagram .....</b>	<b>4-1</b>
4.1	Fundamental principles for LDR programming .....	4-4
4.2	The ladder diagram editor .....	4-6
4.2.1	LDR editor preferences .....	4-7
4.2.2	Tagging a selection in the LDR editor .....	4-8
4.2.3	Functions of the LDR editor .....	4-10
4.3	LDR symbols .....	4-19
4.3.1	Symbol in the condition part .....	4-19
4.3.2	Symbol in executive part .....	4-21
4.4	LDR online display .....	4-27
4.4.1	Functions of the LDR online display .....	4-28
4.4.2	Deactivating LDR online display .....	4-30
<b>5.</b>	<b>Fundamental principles of the FST operating system .....</b>	<b>5-1</b>
5.1	Operands (operating resources of the controller) .....	5-4
5.1.1	Retentive operands .....	5-15
5.2	Multi-tasking .....	5-17
5.2.1	Multitasking applications .....	5-18
5.2.2	Modules for controlling program execution .....	5-20
5.3	Start/stop signals .....	5-21
5.4	Run LED .....	5-23
5.5	Error handling .....	5-24
5.5.1	Error handling without error program .....	5-25
5.5.2	Error handling with error program .....	5-26
5.5.3	Special handling of I/O errors .....	5-27
5.5.4	Modules for error handling .....	5-27
5.5.5	Overview of error numbers .....	5-28

5.6	The Command Interpreter (CI) .....	5-31
5.6.1	Connection to a dialog device .....	5-31
5.6.2	Selecting the command interpreter (Login) .....	5-33
5.6.3	Exiting the command interpreter .....	5-35
5.6.4	CI command .....	5-35
5.6.5	Displaying operands and statuses with Display (D) .....	5-39
5.6.6	Changing operands with Modify (M) .....	5-43
5.6.7	Commands for program controller .....	5-46
5.6.8	Commands for forcing inputs and outputs .....	5-48
5.6.9	Initialising user memory .....	5-50
5.6.10	Password .....	5-51
5.6.11	Driver-specific commands .....	5-52
5.6.12	Linking CI commands .....	5-53
5.7	Miscellaneous .....	5-55
5.7.1	Memory for project files and drivers .....	5-55
5.7.2	Working memory for programs and drivers .....	5-56
<b>A.</b>	<b>Operations (STL and LDR) .....</b>	<b>A-1</b>
A.1	Single-bit operations .....	A-4
A.1.1	Set/Reset (SET, RESET) .....	A-4
A.1.2	Swapping operand and single-bit accumulator (SHIFT, STL only) .	A-6
A.1.3	Assigning single bit value .....	A-7
A.1.4	NOP (No operation) .....	A-9
A.1.5	Logical linking of bits (N, AND, OR, EXOR) .....	A-10
A.2	Multi-bit operations .....	A-16
A.2.1	Loading value to the multi-bit accumulator (LOAD, STL only) ....	A-16
A.2.2	Transferring value from the multi-bit accumulator (TO, STL only) .	A-17
A.2.3	Transferring value direct (LOAD TO, LDR only) .....	A-17
A.2.4	Arithmetical operations (+, -, *, /, <, <=, =, >=, >, <>, INC, DEC) ....	A-18
A.2.5	Logical linking of words (AND, OR, EXOR) .....	A-22
A.2.6	Conversion (SWAP, BID, DEB) .....	A-26
A.2.7	Bit shift operations (SHL, SHR, ROL, ROR) .....	A-30
A.2.8	1's and 2's complement (INV, CPL) .....	A-33
A.3	Standard counter (C...) .....	A-37

A.4	Timer (T...) .....	A-44
A.5	Module call (CMP, CFM) .....	A-52
A.6	Jump (JMP TO) .....	A-54
<b>B.</b>	<b>Menu commands .....</b>	<b>B-1</b>
B.1	Commands in the menu bar .....	B-3
<b>C.</b>	<b>Index .....</b>	<b>C-1</b>
C.1	Index .....	C-3



## Designated use

This software package enables any user familiar with PLC/IPC drives to configure, program and commission the SPS/IPC supported by the software package.

Observe also the standards specified in the relevant chapters, as well as national and local laws and technical regulations.

## Safety regulations

When commissioning and programming, the safety regulations listed in this manual and in the documentation for the control and other components used, must always be observed.

The user must ensure that nobody has access to the positioning range of the connected actuators. The danger area must be made inaccessible by means of suitable measures such as protective screening and warning signs.



### Warning

Depending on the function of the machine/system, the manipulation of signal states can cause serious physical injury and damage to property.

- Exercise extreme caution when using the Change function in order to avoid damage.



### Caution

Incorrect parameterising can cause physical injury and damage to property.

- When setting the parameters, always observe the instructions in the CPX system description and/or the description for each device or module.



### Caution

The command interpreter (CI) contains commands that reorganise or delete parts of the memory. This destroys existing data.

- Only use CI commands if you know their effects!



### Note

The online display always shows the signal status valid in the process image. You should therefore observe the following when Forcing:

- Forced input statuses are transferred to the process image and therefore detected by the control. These are visible in the online display.
- Forced output statuses are **not** transferred to the process image and therefore **not** detected by the control. They are therefore **not** shown in the online display.



### Note

If you globally disable or enable forces or globally delete the channel-oriented force settings (Force table), all signal statuses constrained by forces may also be rendered invalid or valid respectively.

## **Intended readership**

This manual is intended exclusively for technicians trained in control and automation technology who have experience in installing, commissioning, programming and diagnosing PLC/IPCs.

## **Service**

Please consult your local Festo service centre if you have any technical problems.

## **Notes on the use of this manual**

This manual contains specific information on installation, commissioning, programming and diagnosis.

Special information on the supported PLC/IPCs can be found in the hardware documentation for the relevant product.

## Important user instructions

### Danger categories

This manual contains instructions on the possible dangers which may occur if the product is not used correctly. These instructions are marked (Warning, Caution, etc.), printed on a shaded background and marked additionally with a pictogram. A distinction is made between the following danger warnings:



#### **Warning**

This means that failure to observe this instruction may result in serious personal injury or damage to property.



#### **Caution**

This means that failure to observe this instruction may result in personal injury or damage to property.



#### **Please note**

This means that failure to observe this instruction may result in damage to property.

The following pictogram marks passages in the text which describe activities with electrostatically sensitive components.



Electrostatically sensitive components may be damaged if they are not handled correctly.

## Marking special information

The following pictograms mark passages in the text containing special information.

### Pictograms



Information:  
Recommendations, tips and references to other sources of information.



Accessories:  
Information on necessary or sensible accessories for the Festo product.



Environment:  
Information on environment-friendly use of Festo products.

### Text markings

- The bullet indicates activities which may be carried out in any order.
- 1. Figures denote activities which must be carried out in the numerical order specified.
- Hyphens indicate general activities.

## Current information on FST 4.1



### Note

Please note the current information is in the file README.TXT in the main directory on the CD-ROM.

This manual contains the information necessary for operating the FST software, Version 4.1x. FST Version 4.1x is downwards compatible and supports the following PLC/IPC:

- CPX terminal with integrated FEC
- FEC Standard
- FEC Compact
- PS1.

### What's new compared to FST 4.02?

- Supports:
  - CPX terminals with integrated FEC
  - FED Designer
- Contact plan editor
- New drivers and modules.



Further information on the new functions can also be found in Section 1.3.



## Help

Function key F1 takes you straight to a context-based Help page.



After you press SHIFT+ F1 together or click on the adjacent symbols in the tool bar, the mouse pointer becomes an arrow and question mark. You can then click an object for which you require help in the FST program window.

## Contents of the software package:

This software package consists of:


- a CD-ROM,
- the user rights,
- this manual.

## Requirements for the PC

The following is required to operate the software:

- Pentium®-compatible PC with:
- Windows® 95, 98, NT 4.0, ME, 2000 or XP operating system
- Windows 9x: 16 MB RAM recommended, other Windows versions: 64 MB RAM recommended,
- 30 MB hard-drive memory,
- CD-ROM drive,
- a spare serial port for connecting the PLC/IPC (for online mode only).

## Conventions

[File] [New...]	Menu entries appear in square brackets, e.g. the command [Project new ...] opens a new project in the menu [File].
“Abort”, “Cancel”	Names of windows, dialogue windows and buttons, e.g. “Message window”, “Define project file”, “Abort” or “Cancel” as well as designations e.g. “DGPL-PPV-A-KF-B” are shown in inverted commas.
STRG	Names of keys on the PC keyboard are shown in upper case letters in the text. (e.g. ENTER, CTRL, C, F1, etc.).
CTRL + C	<p>Some functions require two keys to be pressed simultaneously. Hold e.g. the CTRL key down and press the C key as well. This is written in the text as CTRL+C.</p> <p>Instructions in this manual to “Click” or “Double click” always refer to the left mouse button. Express reference is made if the right mouse is to be used.</p>
	A large number of functions in FST can be run by clicking a symbol in the tool bar. This symbol will appear next to the text (e.g. the “Allocation list” symbol).
IF I0.0 THEN	Programming examples are marked in a different font, e.g. “ <b>IF I.0.0 SET O0.0</b> ”.



## Product-specific terms and abbreviations

Term/abbreviation	Meaning
Allocation list	List of all operands used in the project. A symbolic operand name and comment can be entered for each absolute operand.
Command Interpreter (CI)	Part of the PLC operating system which accepts and processes commands via a communication interface.
Drivers	Configurable operating system extensions for supporting special devices or functions that can be loaded into the PLC/IPC in dependence of project-specific requirements.
Forcing	This function enables the manipulation of signal states independent of actual operating conditions. The Force mode is used mainly during the commissioning phase for setting certain signals to the desired state for test purposes, even if the circuitry is not available.
Executive part	STL program that contains only one executive part of an incomplete sentence. Conditions that start with IF and an executive part that starts with THEN are omitted.
High byte	High-value byte (left)
I	Digital input
I/O modules	Collective term for modules which provide digital inputs and outputs (input modules and output modules)
I/Os	Digital inputs and outputs
I/O scripts	Files that enable the hardware configuration of the controller in use to be replicated.
Ladder diagram (LDR)	Graphic-based PLC programming language for which the program instructions are based on the principle of electrical circuits.
Parallel logic program	STL program consisting of pure STL sentences, which is completely processed in cycles (like a standard LDR program).
Low byte	Low-value byte (right)
Multitasking	Describes the ability of a processor to process several tasks more or less simultaneously.
O	Digital output

<b>Term/abbreviation</b>	<b>Meaning</b>
Operands	Operating resources and internal memory elements of a PLC/IPC. These include inputs, outputs, flags, indexes etc.
PLC/IPC	Programmable Logic Controller/Industrial PC
Positive edge	Signal change from 0 to 1
Process image	The process image is part of a controller's system memory. At the start of the cyclical program, the signal states of the input assemblies are transferred to the process image for the inputs. At the end of the cyclical program, the process image for the outputs are transferred to the output assemblies as the signal state.
Real-time clock	Independent of the supply voltage to the PLC/IPC and always delivers the correct time and date.
Runtime library	Contains several I/O scripts and drivers for the supported controller types.
Step program	STL program in which one or several STL sentences are grouped and structured into individual steps. Program branches (jumps) enable certain steps to be "skipped".
STL	Text-based PLC programming language. Programming instructions can be created as a structured, step program or in a flexible sequence of instructions as a parallel logic program or executive part.

Tab. 0/1: Product-specific terms and abbreviations

# **Installation and general operator instructions**

## **Chapter 1**

Contents

**1. Installation and general operator instructions ..... 1-1**

1.1 FST software package ..... 1-4

1.1.1 Installing FST ..... 1-4

1.1.2 Deinstalling FST ..... 1-6

1.2 Starting FST ..... 1-6

1.3 What’s new ..... 1-7

1.4 The FST operating interface ..... 1-8

1.4.1 FST features ..... 1-10

1.4.2 Window within the FST ..... 1-14

1.4.3 Program editors ..... 1-16

1.4.4 Multiple-column list fields in dialog windows ..... 1-18

1.4.5 The FST software Help system ..... 1-19

1.4.6 Exiting FST ..... 1-21

## 1. Installation and general operator instructions

Contents of this chapter	<p>This chapter provides an overview of the components and functionality of the FST software. It contains basic information on:</p> <ul style="list-style-type: none"><li>– installing and deinstalling the software package</li><li>– setting up the directory paths</li><li>– the special features of FST</li><li>– the general operation of FST</li><li>– the FST Help system.</li></ul>
Further information	<p>Please note the latest information contained in the README.TXT file in the installation directory of the FST software.</p>

## 1. Installation and general operator instructions

### 1.1 FST software package

The software package will support your configuration, programming and commissioning of the following devices:

- CPX terminal with integrated Front End Controller
- FEC Compact
- FEC Standard
- PS1 Professional

#### 1.1.1 Installing FST

The FST software package can be set-up on your PC in line with your specific requirements. You can:

- install FST in the language of your choice,
- install example files
- deinstall FST.



#### **Note**

If a different version of FST is already installed, deinstall this software before installing the new version (see Section 1.1.2).

Proceed with installation as follows:

1. Insert the FST installation CD-ROM.
2. Change to the CD ROM drive, e.g. in Explorer.
3. Double-click to start Setup.exe.
4. Select the language in which you want to install FST and confirm your selection with [OK].

## 1. Installation and general operator instructions

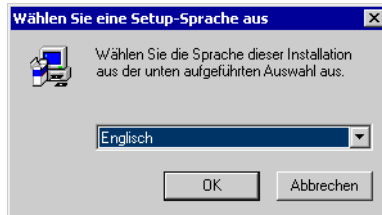


Fig. 1/1: Selecting the dialog window language

Follow the instructions in the installation program. [Forward] moves you to the next step, with [Back] takes you back to the previous step:

- Welcome
- Display the terms of use and current information on FST
- Select the installation directory
- Select the project directory
- Select setup type
  - Standard installation (all components)
  - Minimal installation (required program files only)
  - User-defined installation (components are freely selectable)
- Select the program folder
- Display successful installation and complete the installation program

When installation is completed, you will find the FST entry in the start menu under [Festo Software].

## 1. Installation and general operator instructions

### 1.1.2 Deinstalling FST

FST can be deinstalled using the [Software] function in the Windows system control.

Proceed here as described in your Windows manual.

## 1.2 Starting FST

How to start FST:

- From the Start menu, select [Programs] [Festo Software] [FST4...] [FST 4]. The FST program window then appears (see also Section 1.4.)

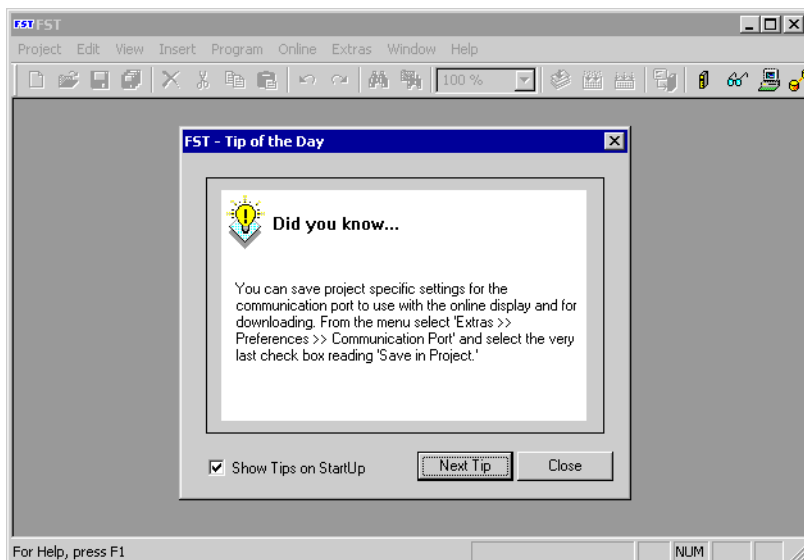


Fig. 1/2: Starting the FST Software



## 1. Installation and general operator instructions

### 1.3 What's new

New compared to FST 4.02	
Devices and external tools that are also supported	<ul style="list-style-type: none"><li>– CPX terminal with integrated Front End Controller is supported</li><li>– FED Designer</li></ul>
Commissioning	<p>With the CPX terminal:</p> <ul style="list-style-type: none"><li>– Configuration, parameterisation, reference/actual comparison, diagnosis etc.</li><li>– Graphic representation of reference and actual configuration</li></ul>
Programming	<ul style="list-style-type: none"><li>– Programming is now also possible in ladder diagram</li><li>– Parameter description also for module outputs</li><li>– Possible to create module-specific parameter description files</li><li>– Global find function (finds operands in all programs)</li><li>– Allocation list: A CSV file that can be used by FED Designer is created at each save.</li></ul>
Diagnosis	<ul style="list-style-type: none"><li>– Support for breakpoints (provided the runtime system supports the connected control)</li></ul>
Drivers	<ul style="list-style-type: none"><li>– New drivers (e.g. B. FECCPX, STEPLITE, WATCHDOG, FPMATHDR, WEB_SRVR, SMTP)</li><li>– New general modules (e.g. INRANGE, MINMAX, SCALE)</li><li>– New modules (e.g. OPENCOMX, modules for access to parameters and data of the CPX terminal)</li></ul>
Miscellaneous	<ul style="list-style-type: none"><li>– Integrated Web Browser for displaying the Start page of the connected control (uses MS Internet Explorer functions)</li><li>– Search function for simple selection of IP address</li><li>– Projects can be conveniently sent by email</li><li>– Message window: is now a docking window</li></ul>

Tab. 1/1: New compared to FST 4.02



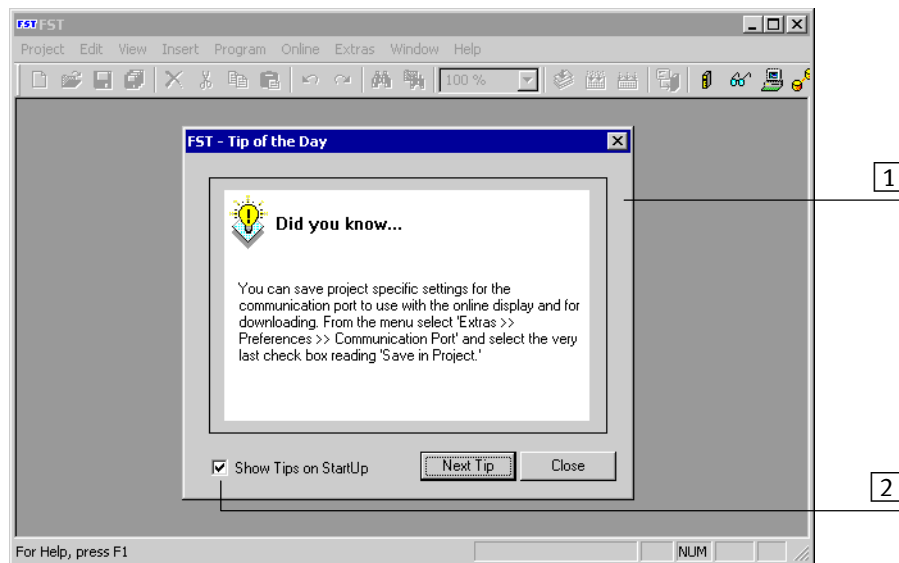
Further information can be found in the README.TXT file.

## 1. Installation and general operator instructions

### 1.4 The FST operating interface

When FST is started, the FST program window appears. First, a logo appears in the foreground which is then automatically hidden after a few seconds. Click on the logo to close it immediately.

The “Tip of the Day” window is then shown. In the bottom section of the window you will see the “Show Tips after on StartUp” checkbox. Tick to stop the tips appearing.



1 “Tip of the Day” window

2 Checkboxes

Fig. 1/3: Operating interface of the FST software

FST uses what is referred to as the multiple document interface (MDI). A separate window within the FST program window opens for each document. The document window can be activated and arranged using the commands in the “Window” menu.

## 1. Installation and general operator instructions

The size and position of the windows is saved between the FST sessions. If the screen resolution is changed, Windows adjusts the size and position of the windows.



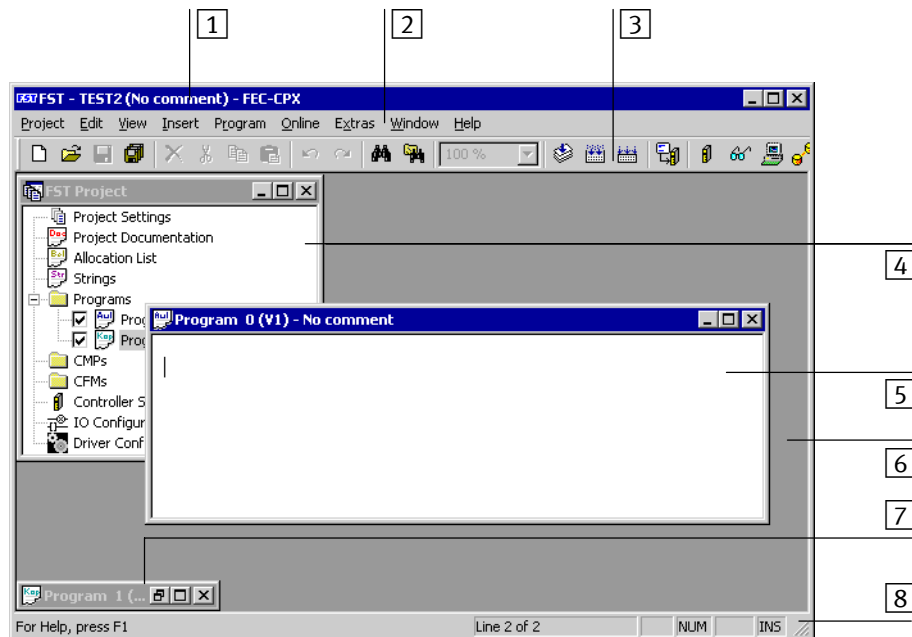
The FST software package is an application for the Windows operating system. As such, the program interface and operation are consistent with the usual Windows standard. The buttons, menu bar, picture scroll bars etc. of the FST software therefore behave as they do in most other Windows-based programs. General information on operating Windows applications can be found in your Windows software documentation.

The following sections provide a brief overview of the special features in FST software.

## 1. Installation and general operator instructions

### 1.4.1 FST features

The FST program window contains the following features:



- |                  |                         |
|------------------|-------------------------|
| 1 Title bar      | 5 Program editor window |
| 2 Menu bar       | 6 Workspace             |
| 3 Toolbar        | 7 Reduce to symbol      |
| 4 Project window | 8 Status Bar            |

Fig. 1/4: Operating interface of the FST software

## 1. Installation and general operator instructions

**Title bar** As with most Windows applications, the left side of the title bar contains the system menu field that you can use to open the system menu corresponding to the window. The right side of the title bar contains the symbols “Minimize”, “Maximize” and “Close”. You can use these symbols to reduce the FST program window to its symbol, toggle between full image and window display or finish working with the FST software.

**Menu bar** The menu bar is located at the top edge of the FST program window below the title bar. Clicking on the menu names opens the menu. Alternatively, you can open a menu by pressing the ALT key and the corresponding code letter at the same time. This menu is underlined in the menu title. The menus offer the following commands:

<b>Menu</b>	<b>Brief description</b>
[Project]	Commands for working with projects; closing FST
[Edit]	Undo, clipboard commands (cut, copy etc.), find and replace, select all
[View]	Opens various internal editors such as record list editor, I/O configuration, PLC settings, displays message window, shows and hides various symbol bars
[Insert]	Inserts new programs and modules into the project and new entries into the active window, e.g. in the driver configuration
[Program]	Manages control programs and prints the contents of the active window
[Online]	Commands for communication between FST and the connected control
[Extras]	Changes FST settings, manages the FST library, configures and selects external tools
[Window]	Commands for organising windows within the FST software
[Help]	Online Help, Tip of the Day, Info. on FST

Tab. 1/2: Menus in the menu bar

## 1. Installation and general operator instructions

### Toolbar

The standard position of the icon bar is below the menu bar. However, it can be also be dragged onto the workspace as a separate window.

The icon bar enables many functions to be accessed using the mouse. Clicking on a symbol runs the required command. As the mouse pointer is moved over a symbol, its function is displayed in the status bar. If you want to see the name of an icon bar button, hold the cursor on the button. The name is shown just under the mouse pointer.



Fig. 1/5: Toolbar as a separate window



The [View] [Toolbar] command shows or hides the icon bar. A tick next to the menu entry indicates whether the icon bar is visible.

### Workspace

In this area, you can work with the various windows of the FST software.

### Project window

In the project window, you can manage and configure all features of your control project.

### Program window

You can edit your programs by opening the corresponding program window.

### Icon for a window

To improve the overview, you can also reduce individual windows in the workspace to an icon.

## 1. Installation and general operator instructions

### The status bar

#### Status bar

The status line is located at the top edge of the FST program window. The status bar contains the following information:



- 1** Legends or description of the menu command or symbol under the mouse pointer
- 2** Number of the current line and number of lines or entries in the active document
- 3** These areas indicate the active key functions (see Table below)

Fig. 1/6: Status bar

Display	Description	Key
CAP	Shift lock active	SHIFT LOCK KEY
NUM	Number lock active	NUM LOCK
SCRL	Scroll active	SCROLL
INS	Overstrike mode active	INSERT

Tab. 1/3: Displayed key functions



The [View] [Status Bar] command shows or hides the status line. A tick next to the menu entry indicates whether the status line is visible.

1. Installation and general operator instructions

1.4.2 Window within the FST

A separate window within the FST program window opens for each document. This document window can be managed using the commands in the [Window] menu.

Commands in the [Window] menu	Description
[Close]	Closes the active document and/or program window. If the document contains unsaved changes in the active window, the relevant prompt will appear. You can also close FST windows using the “Close” button in the title bar of the corresponding window.
[Cascade]	Cascades the document windows behind each other.
[Tile Horizontal]	Splits the document window equally from top to bottom
[Tile Vertical]	Splits the document window equally from left to right
[Arrange Icons]	Minimises the windows to icons at the lower edge of the FST program window.
[1 ..., 2 ..., 3 ... ]	A list of names for the opened windows is shown in the bottom section of the menu [window]. A tick indicates which window is active. Clicking on a window name activates that particular window.

Tab. 1/4: Menu commands [Window]

- The window edge

Position the mouse pointer at an edge or corner of the window and it will turn into a double arrow. You can now hold the left mouse button down and move the edge or the corner of the window to change its size.
- Picture scroll bars

The picture scroll bars are located at the bottom and right edges of a window. They are shown only if the window content will not fit into the workspace for this window size.



## 1. Installation and general operator instructions

The picture scroll bar shows the horizontal and vertical position of the window content. Move the position of the window content by holding the mouse button down or clicking on the buttons.

### Context menus

In FST windows, commands can also be selected through what are referred to as “Context menus”. A context menu appears if you right-click on an element or within a window.

Context menus are matched to the selected element or the corresponding window content. You can use context menus in many areas in the FST. This option is not always described in detail. Simply try out the function.

## 1. Installation and general operator instructions

### 1.4.3 Program editors

A control program can be edited in an open program window. The program window is therefore also referred to as program editor. The FST software provides the following program editors:

- Ladder diagram (LDR editor)
- Statement list editor (STL editor)

You can create your control programs using these program editors. When you open a program, its contents will appear in a program editor window within the FST program window. You can open and display as many programs as you like.



To toggle between programs, open the [Window] menu. All open windows are displayed as menu entries. Select one of the entries to open the desired window.

Each editor offers special functions and editor help windows, e.g. a special fast input bar. Detailed information on the STL editor be found in chapter 3. Information on the LDR editor be found in chapter 4. The following general editor commands in the [Edit] menu can be used for both editors:

## 1. Installation and general operator instructions

Commands in the [Edit] menu	Description
[Undo]	Cancels the last changes to a document in the usual manner.
[Redo]	Restores the original status.
[Cut] <sup>1)</sup>	Deletes selected objects from the current window and places them on the clipboard. The previous clipboard content is lost.
[Copy] <sup>1)</sup>	Copies selected objects from the current document to the clipboard.
[Paste] <sup>2)</sup>	Inserts the contents of the clipboard to the current document at the selected position.
[Delete] <sup>1)</sup>	Deletes selected objects from the current document. The data are <b>not</b> copied to the clipboard.
[Find...]	Opens the "Find..." dialog that enables you to search for text in the current document.
[Global find...]	Opens the "Global find..." dialog that enables you to search for text throughout the entire project.
[Find next]	Repeats the last search.
[Replace...]	Opens the "Replace..." window that enables you to replace text in the current document.
[Select all]	Selects the entire content of the document.
<sup>1)</sup> The command is active if no item is selected. <sup>2)</sup> The command is inactive if the clipboard is empty.	

Tab. 1/5: General commands in the [Edit] menu

## 1. Installation and general operator instructions

### 1.4.4 Multiple-column list fields in dialog windows

Some dialog windows in the FST software feature multiple-column list fields, as shown in the diagram below:

- 1 Multiple-column list field
- 2 Column header
- 3 Buttons with other commands

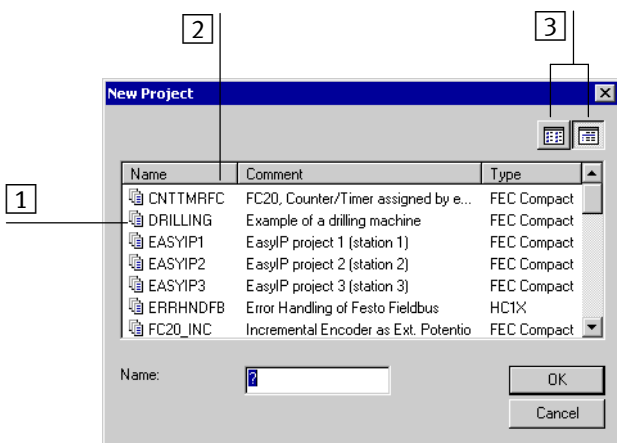


Fig. 1/7: Multiple-column list field (example)

Multiple-column list fields offer the following functions:

- The column width can be adjusted in the table header using the mouse.
- Each column is sorted alphabetically by clicking on the column header. Clicking again reverses the sorting sequence.
- The display format of the list field can be set by clicking on the corresponding button in the top right-hand corner.

## 1. Installation and general operator instructions

### 1.4.5 The FST software Help system

Operation of the FST software is described in the Online Help. This enables you to use the benefits and functions of Windows Help systems such as context-sensitive Help, keyword search, cross-references etc. The content of this manual is a salient part of the Help system.

#### Structure of the Help system

The components of the Help system are shown in the [Contents] tab when the Help system is selected using the [Help] [Help Topics] command.

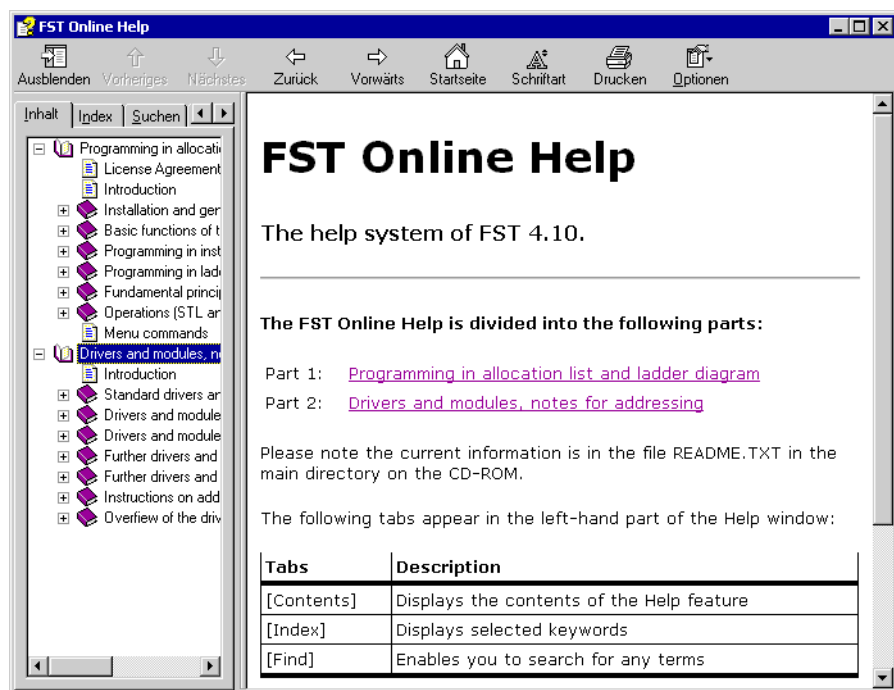


Fig. 1/8: The FST Help system

## 1. Installation and general operator instructions

### [Help] menu

Information on various topics can be selected using the following commands in the [Help] menu:

- Use the [Help Topics] command to open the start page of FST Online Help. From this start page, you can open Help pages containing detailed operating instructions or FST reference data.
- The following tabs appear in the left-hand part of the Help window (can be shown or hidden as required using [Help Topics]):

<b>Tabs</b>	<b>Description</b>
[Contents]	Displays the contents of the Help feature
[Index]	Displays selected keywords
[Find]	Enables you to search for any terms

- The [Tip of the Day] command gives you a brief tip.
  - Click on the “Next tip” button to view another tip.
  - Close the window by clicking on the “Close” button.
- If you have an Internet connection, you can access additional online information on Festo using the [Internet] command.
- You can obtain the copyright notice and version number of your FST using the [About FST] command.

## 1. Installation and general operator instructions

### Using the Help system



Function key F1 takes you straight to a context-based Help page.



After you press SHIFT+F1 together or click on the adjacent symbols in the tool bar, the mouse pointer becomes an arrow and question mark. You can then click an object on which you require help in the FST program window.



The Help system is used in the same way as other Windows Help files. Further information can therefore be found in your Windows manual.

#### 1.4.6 Exiting FST

To finish your FST session, select the [Exit] command in the [Project] menu. Alternatively, you can also use the [Close] command in the system menu of the FST application or click the “Close” button in the FST program window.

## 1. Installation and general operator instructions



# **Basic functions of the FST software**

## **Chapter 2**

## Contents

<b>2.</b>	<b>Basic functions of the FST software .....</b>	<b>2-1</b>
2.1	Organising projects .....	2-5
2.1.1	The project directory .....	2-6
2.1.2	General project information .....	2-9
2.2	Working with projects .....	2-12
2.2.1	Creating a new project .....	2-12
2.2.2	Changing the project properties .....	2-15
2.2.3	Opening a project .....	2-16
2.2.4	Closing the current project .....	2-17
2.2.5	Managing projects (delete, copy, rename...) .....	2-18
2.2.6	Preparing the project for importing .....	2-20
2.2.7	Downloading projects into the controller .....	2-21
2.2.8	Updating projects in the controller .....	2-22
2.2.9	Uploading project sources from the controller .....	2-23
2.2.10	Displaying the project file content .....	2-24
2.2.11	Deleting the restorable files .....	2-25
2.2.12	The project documentation .....	2-25
2.2.13	Printing a project .....	2-26
2.2.14	Archiving or restoring projects .....	2-30
2.3	Replicating the hardware configuration .....	2-31
2.3.1	Hardware configuration for PS1, FEC Standard and FEC Compact .	2-32
2.3.2	Hardware configuration for CPX terminal (CPX-FEC) .....	2-36
2.3.3	Online and offline mode of the hardware configurator (CPX-FEC) .	2-38
2.3.4	Save the actual configuration as the nominal configuration (CPX-FEX) .....	2-39
2.3.5	Performing nominal/actual comparison (CPX-FEC) .....	2-41
2.3.6	Manually generating the nominal configuration (CPX-FEC) .....	2-43
2.3.7	Setting and changing the address mapping (CPX-FEC) .....	2-48
2.3.8	Parameterising the CPX terminal with FST (CPX-FEC) .....	2-48
2.3.9	Trace memory .....	2-54
2.4	The driver configuration .....	2-55

## 2. Basic functions of the FST software

2.4.1	Editing functions for driver configuration .....	2-56
2.5	Controller settings (PLC settings) .....	2-60
2.5.1	Settings for runtime behaviour .....	2-60
2.5.2	Drives .....	2-62
2.5.3	Options .....	2-64
2.5.4	Password protection .....	2-65
2.5.5	Settings for downloading .....	2-67
2.6	The allocation list .....	2-69
2.6.1	Functions of the allocation list editor .....	2-72
2.7	The Strings Editor .....	2-75
2.7.1	Functions of the Strings Editor .....	2-76
2.8	Managing the controller programs .....	2-78
2.8.1	Creating a new program or module .....	2-79
2.8.2	Importing programs and modules .....	2-81
2.8.3	Exporting programs and modules into the library .....	2-84
2.8.4	Viewing and changing program and module properties .....	2-85
2.8.5	Deleting programs or modules .....	2-86
2.8.6	Opening a program or module .....	2-88
2.8.7	Compiling programs and modules .....	2-89
2.8.8	Selecting an object for compilation and downloading .....	2-90
2.9	Online mode .....	2-91
2.9.1	Configuring the online link .....	2-92
2.9.2	The online control panel .....	2-95
2.9.3	Setting password through the online control panel .....	2-98
2.9.4	The online display .....	2-99
2.9.5	FST file transfer .....	2-109
2.9.6	The command interpreter terminal .....	2-113
2.9.7	Online mode of the hardware configurator (FEC-CPX only) .....	2-114
2.9.8	STL and LDR Online display .....	2-114
2.9.9	The Web Browser .....	2-115
2.10	Runtime library (drivers and I/O scripts) .....	2-117
2.10.1	Managing I/O scripts in the runtime library .....	2-119
2.10.2	Managing drivers in the runtime library .....	2-121
2.11	External Tools .....	2-124
2.11.1	Inserting or changing a program call .....	2-125

## 2. Basic functions of the FST software

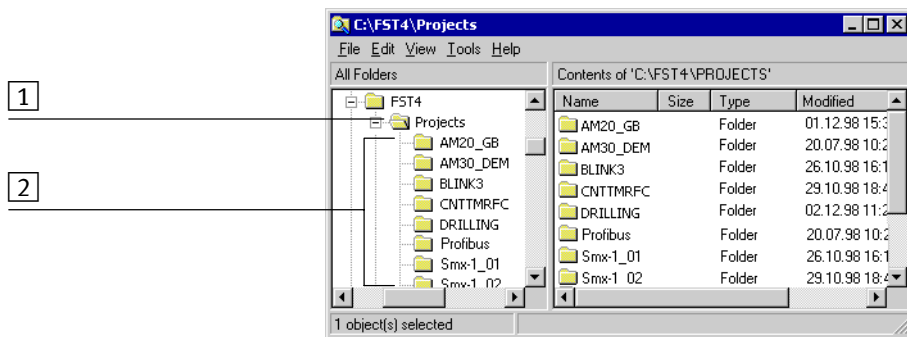
Contents of this chapter	<p>This chapter provides an overview of the basic functions featured in the FST software.</p> <p>FST's integrated project manager enables you to edit the applications and backup your data, regardless of location.</p> <p>A project consists of the programs and data for your control application. They correspond to a specific problem to be solved using the project. The term "Project" used in FST therefore differs in some aspects from the definitions applicable to conventional "Technical Planning" and "Project Management".</p>
Further information	<p>Information on creating control programs can be found in chapters 3 and 4 and also in Appendix A.</p> <p>Fundamental principles of the FST operating system can be found in chapter 5.</p>

### 2.1 Organising projects

The means employed to solve the controller task are summarised in FST in the form of a project. The project comprises:

- the source of the programs
- modules imported from the library
- the I/O configuration
- the driver configuration
- the allocation list
- the project-specific settings.

All components of a project are stored in a separate directory (folder) within the higher-order project directory. The path of the higher-order project directory is specified during installation. You can, however, change this path at any time. This enables practical grouping of a large number of projects.



1 Higher-order project directory

2 Directory of individual projects

Fig. 2/1: Example of a project directory (Windows Explorer)

## 2. Basic functions of the FST software

### 2.1.1 The project directory

The project directory is a higher-order directory (folder) to which the individual FST projects are saved in their own directories. When a project is opened, only projects in the current project directory are offered for selection. The project directory can be located on your hard drive, on a network drive or even on a removable storage medium.



Configure  
project directory

When a new higher-order project directory is selected or created, any project already open automatically closes.

Proceed as follows to set the directory path for the higher-order project directory and apply general FST settings:

1. Select [Extras] [FST Preferences...]. The “FST Preferences” dialog window will be displayed.

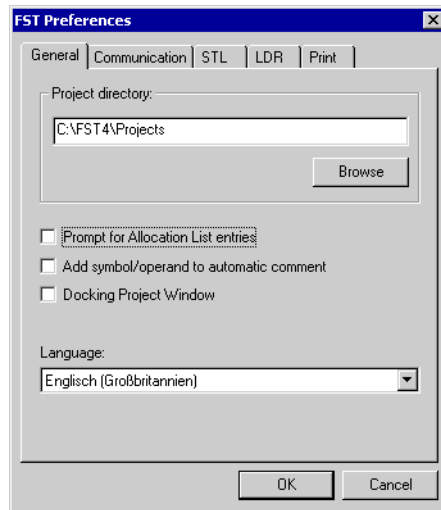


Fig. 2/2: FST Preferences

The “General” tab offers the following settings:

## 2. Basic functions of the FST software

Field	Description
Project directory	Higher-order directory (folder) to which the individual FST projects are saved in their own directories.
Prompt for Allocation List entries	When a new operand is entered into a LDR or STL program, a window opens to enable it to be entered into the record list at the same time.
Add symbol/operand to automatic comment	Indicates the symbolic name of the operand and the allocation list comment in the LDR and STL program.
Docking Project Window <sup>1)</sup>	Enables the project window to be fixed to the left or right side of the FST program window
Language <sup>1)</sup>	Offers the possible FST dialog languages for selection
<sup>1)</sup> FST must be restarted after these settings are changed.	

Tab. 2/1: “General” tab field

2. Enter the required directory path into the “Project directory” field. Or click the “Browse” button to select an existing directory. Please observe the following:



### Note

If you wish to use DOS Tools, e.g. FST fieldbus configurators, the project directory must not contain long file names (max. 8 characters) or spaces.



### Note

Projects created for FST 3.x cannot be used until converted to FST 4.x. You should therefore use a directory that does not contain projects created for FST 3.x.

## 2. Basic functions of the FST software

The higher-order project directory must not be a sub-directory of the FST installation directory. We recommend that you use the higher-order project directory only for FST projects.

You can configure a sub-directory of the FST installation directory by entering the relative path (starting with a dot). Example:

• \PROJECTS

3. Apply any necessary further settings (see Tab. 2/1) and confirm your entries with “OK”. The project directory is then selected and/or created and any further necessary settings applied.



2.1.2 General project information

FST supports various IPC controllers. The controllers require various different drivers and modules for the same task.

To ensure that the project contains the correct drivers and modules for the target controller, each project must be assigned a controller type. A project can be loaded only into the controller for which it was created. The controller type is assigned to the project when it is created, although it may subsequently be changed.



Some parts of the project (e.g. driver configuration) may occur several times for various controller types. The part of the current controller type is always edited.

Same for all controller types	Different for each controller type
<ul style="list-style-type: none"><li>– Project name and comment</li><li>– Sources of the programs</li><li>– Project documentation</li><li>– Allocation list</li></ul>	<ul style="list-style-type: none"><li>– Modules imported (compiled) from the library</li><li>– IO configuration</li><li>– Driver configuration</li><li>– Controller settings</li></ul>

Tab. 2/2: Components of a project

## 2. Basic functions of the FST software

### The project window

An open project is displayed in the project window. The project window provides an overview of all important parts of the project. Double-click to edit a project part and select other important functions via its context menu.

The project window is shown immediately after a project is created. It can be closed like any other window. It is reopened and/or placed in the foreground using the [View] [Project window] menu command.

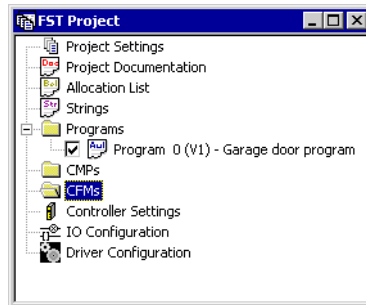


Fig. 2/3: Project window

The project window always has the same branches. The “Programs”, “CMPs” and “CFMs” folders contain a project’s programs and modules and the number of items in these folders therefore differs from project to project.

## 2. Basic functions of the FST software

The following accesses to programs are possible via the project window:

- Double-click on the program if you want to open the source for editing. Imported modules have no source in the project, double-click to open the properties window for the module.
- The properties window can be opened via the context menu of the program item. In this window, you can change the program type (Program/CMP/CFM), the program number, the version number and the comment. The source programming language cannot be changed.
- To delete a program, select and press the DEL key. A program can also be deleted via its context menu.
- Click on the box in front of the program name to make or delete the program selection. Only the selected programs are integrated into the project file as the project is compiled and then imported into the controller. At the same time, select only one version of the same program.
- The window to add a new program or import a compiled module from the library can be opened via the program context menu or the “Programs”, “CMPs” and “CFMs” folder.

### 2.2 Working with projects

For project management, FST provides various options described in the following Sections:

- Creating or opening projects,
- Managing projects (delete, copy, rename...)
- Printing out complete projects or project parts (programs, allocation lists etc.),
- Loading and/or downloaded projects or project parts.

#### 2.2.1 Creating a new project

You create a new project when you want to program a new controller task. Once created, it is saved to the current project directory.



Before you start to create a new project, setup the required project directory (see also Section 2.1.1). When a new project is created, any project already open automatically closes.

How to create a new project in the current project directory:



1. Select [Project] [New ...]. The “New Project” dialog window will be displayed.

## 2. Basic functions of the FST software

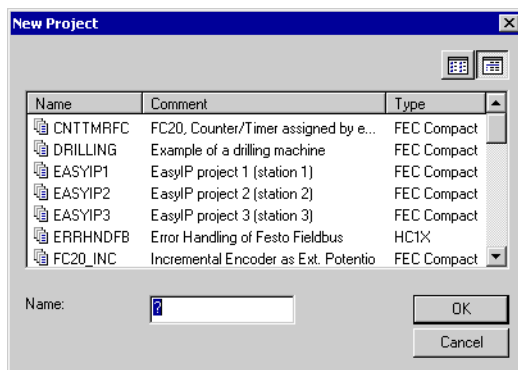


Fig. 2/4: Create new project

The list displays names, comments and controller types of the existing projects.

2. Enter a name for the new project (max. 8 characters) in the “Name” field.



Project names must not contain spaces or the following characters: \ / : \* ? ” < > |.

All other characters are permitted. However, it is advisable to use only letters, figures and underscore.

The project name must not contain more than 8 characters. It is loaded as the project is imported into the controller. Each project name within a project directory must be unique. Entered letters are always interpreted in upper case.

3. Confirm with “OK”. The “Project Settings” dialog window will then open.

## 2. Basic functions of the FST software

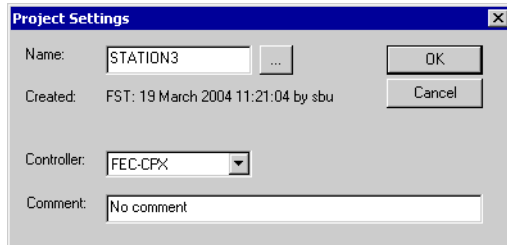


Fig. 2/5: Project Settings

4. Now select the controller type for the new project and enter a comment.
5. Confirm with “OK”. The project is then created below the current project directory (see Section 2.1.1).

Various files such as program sources, documents, allocation list and much more are saved in the project directory. These files are managed by FST and external tools.



### Note

You can also save your own, project-related files in the project directory. You must, however, select file names that are not used by FST as otherwise these files may be overwritten.



### Note

Each project contains in its directory a file named **PROJECT.FW4**, which contains the current project settings (e.g. comment and controller type). Never make manual changes to this file.

2.2.2 Changing the project properties

You can view and change project properties.

How to open the project property dialog window:



Project Settings

- Select [Project] [Settings...] or double-click on “Project Settings” in the project window.

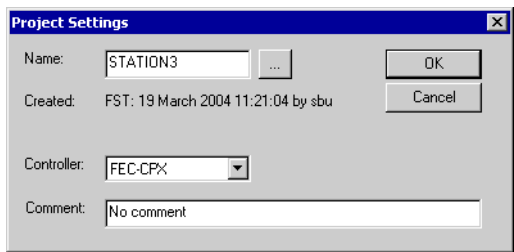


Fig. 2/6: Project Settings

You can set the following:

Field	Description
Name	Project names (max. 8 characters) must not contain spaces or the following characters: \ / : * ? " < >  .
Created	Indicates the date and time on which the project was created and its author.
Controller	Controller type for the project
Comment	To aid project identification, you can enter descriptive text here (max. 255 characters).

Tab. 2/3: Fields in the project properties dialogue window

### 2.2.3 Opening a project

When a new project is opened, any project already open automatically closes. The last 8 edited projects are displayed as a list in the menu [File], from which they can be opened directly. On opening, only those projects present in the current project directory are displayed.



Before you open a new project, set-up the required project directory (see also Section 2.1.1).

How to open a new project from the current project directory:



1. Select [Project] [Open...] The “Open Project” dialog window containing all projects in the current project directory is then displayed.

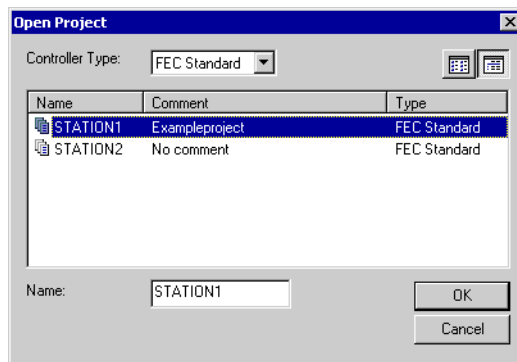


Fig. 2/7: Open Project

2. If you wish to see only those projects created for a particular controller type, select a controller type from the “Controller Type” field.
3. If only project names are to be displayed in the multiple-column list field, click the relevant button in the top right-hand corner to change the display format.



## 2. Basic functions of the FST software

4. Select a project from the multiple column list field. You can also enter the project name in the edit field itself.
5. Confirm with “OK”. The selected project then opens.

The name, comment and controller type of the current project is displayed in the title list of the FST program window.



When FST is started, the last project to be edited opens automatically. Several FST sessions can run at the same time. However, the same project cannot be opened in several sessions at the same time. When a project is opened, a locking file that prevents it from being opened again is created in its directory.

### 2.2.4 Closing the current project



Before a project is opened or a new project is set-up, the current project automatically closes.

How to close an open project:

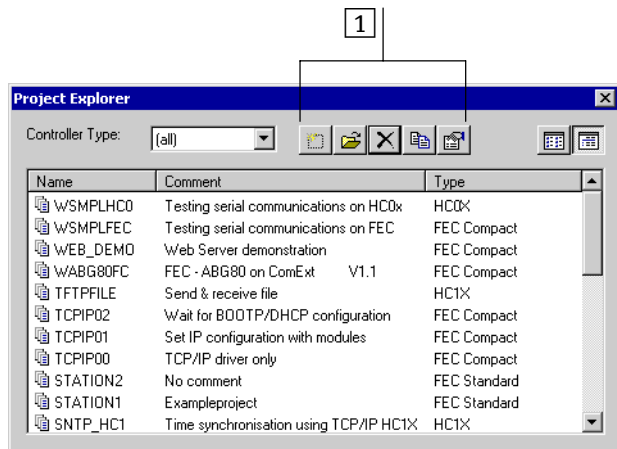
- Select [Project] [Close]. The program will then close.

When a project is closed, the size and position data of most windows are saved. On opening, this data that reflects the status before closing, is restored.

## 2. Basic functions of the FST software

### 2.2.5 Managing projects (delete, copy, rename...)

The [Project] [Explore] menu command makes project management simple and convenient. When a command is selected, the “Project Explorer” dialogue window appears listing all projects for the current project directory. Name, comment and controller type are displayed in columns for each project in the project directory.



#### 1 Buttons for project management






Fig. 2/8: Manage projects

Double-clicking on a project opens the product properties dialog field (see also Fig. 2/6). In this field, you can change project name, controller type and project comment.

If you wish to see only those projects created for a particular controller type, select a controller type from the “Controller Type” field.

## 2. Basic functions of the FST software

Buttons are available for the following operations above the project list:

Icon	Operation
	Setup new project
	Open project
	Delete project
	Copy project. The project is copied under a new name. The properties of the new project must be set.
	Changing project properties. The name, comment and controller type can be changed.

Tab. 2/4: Operation in “Project Explorer” dialog window

## 2. Basic functions of the FST software

### 2.2.6 Preparing the project for importing

To be able to import and run a project in the controller, its components must be converted (compiled) to binary format. The binary files are then integrated (linked) into a file – the project file.

How to prepare a project for downloading:

- To compile only programs that were changed since the last compilation, select [Project] [Make Project].
- To compile all programs, whether changed or not, select [Project] [Build Project]. This command is useful particularly if you have de-archived a project or updated the FST software.

Once the relevant command has been selected, the unsaved programs are automatically saved. The result of the compilation and the link is displayed in the message window.

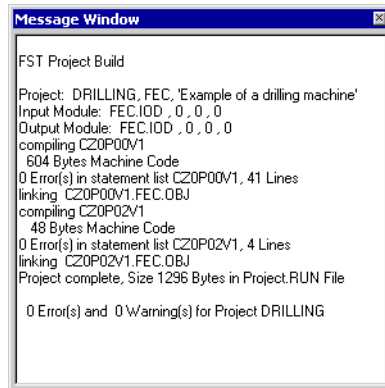


Fig. 2/9: Message Window (example)

If an error occurs during compilation, the operation is cancelled. The error is displayed in bold font in the message window. Double-clicking on an error in the message window takes you directly to the relevant location in the source code. You can then correct the error. Use F4 to skip to the next error.

### 2.2.7 Downloading projects into the controller

To be able to download a project, it must have first been compiled. A project can be downloaded only to the controller type for which it was compiled. Set the required controller type in project settings.



Before projects are downloaded, various defaults must be set. It is possible, for instance, to load into the controller, the sources required to restore the project. Further information on the possible default settings can be found in Section 2.5.

How to download the current project to the controller:



- Select [Online] [Download Project]. All selected and required files are transferred to the controller. The “Download” dialog provides information on the current data transfer status. The message window displays details on the download procedure.

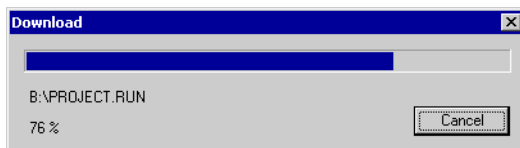


Fig. 2/10: Download

Press “Cancel” to prematurely end the download process.

### 2.2.8 Updating projects in the controller

Instead of downloading the entire project into the controller, it is possible to transfer only those changes made since the last download operation. If projects are large, this update method can be much faster than downloading the entire project.

An update cannot be run until the entire project has been downloaded at least once beforehand.



#### Note

A project update is not possible if:

- programs have been added or removed,
- the driver configuration has been changed.

Update a project in the controller as follows:

- Select [Online] [Update Project]. All required files are transferred to the controller. The “Update” window provides information on the current data transfer status. The message window displays details.

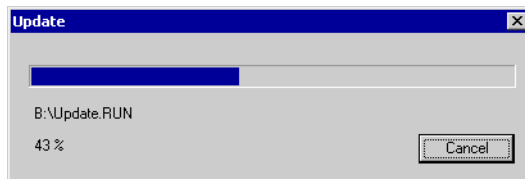


Fig. 2/11: Update

Press “Cancel” to prematurely end the update operation.

## 2. Basic functions of the FST software

### 2.2.9 Uploading project sources from the controller

If a project was downloaded into the controller with the sources included (see also Section 2.5), the project can be restored from these sources.

Restore a project from the sources stored in the controller as follows.

1. Select [Online] [Upload Project]. The sources are then transferred to your PC. The “Upload Source” window provides information on the current data transfer status. The message window displays details.

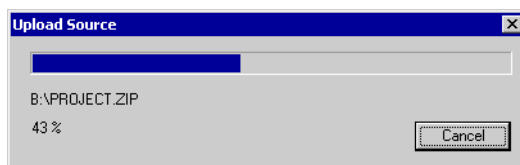


Fig. 2/12: Upload Source

After the upload operation, the “Upload Project” window opens.

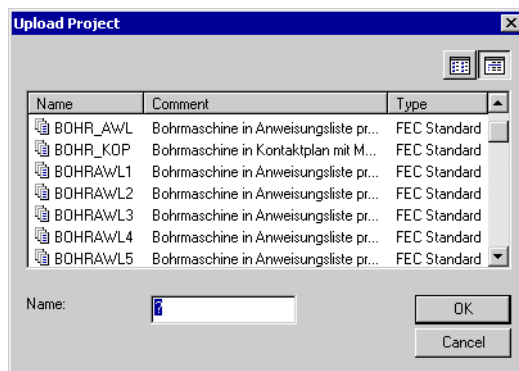


Fig. 2/13: Upload Project

## 2. Basic functions of the FST software

2. Enter a name for the project in the “Name” field and confirm with “OK”.



The naming conventions for creating a project are also applicable here (see Section 2.2.1).

### 2.2.10 Displaying the project file content

When you select the [Project] [List Project File] menu command, the content of the last created project file is displayed in the message window. This file is transferred as the result of preparing the project for downloading (compiling) to the controller. The display provides useful information for troubleshooting.

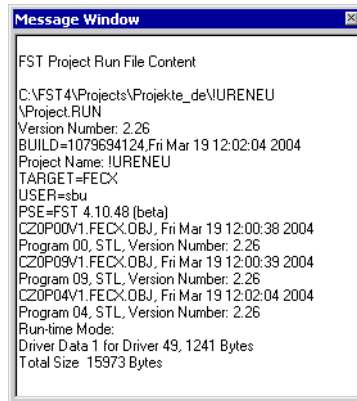


Fig. 2/14: Message Window



The function only becomes available when the project has been prepared (compiled) for download.



### 2.2.11 Deleting the restorable files

As the project is being prepared for download, several temporary files are created. To save space on the hard drive, these files can be deleted.

- To clear a project, select [Project] [Clean Up].

Next time the project is downloaded, all the required files are created. The project is automatically cleared before archiving.

### 2.2.12 The project documentation

When a new project is created, an empty file called **PROJECT.TXT** is created in the project directory. This file is designed for the project documentation. When the project documentation is opened, the editor registered on your PC for this file extension is started. FST does not have its own text editor.

To use a file with a different file extension and therefore a different editor for the project documentation, proceed as follows:

1. Rename the **PROJECT.TXT** file (e.g. as **PROJECT.DOC**, if you wish to use Microsoft Word® for editing) or create a new file using the corresponding editor in the project directory.
2. Open the context menu for the “Project Documentation” entry in the project window and select the [Properties] command.
3. Select the file for the project documentation.



When the project is printed, the **PROJECT.TXT** file, where present, is printed as the project documentation. If you have selected a different file, this must be printed separately using the corresponding editor.

## 2. Basic functions of the FST software



Project Documentation

- To edit project documentation, double-click on “Project Documentation” in the project window or select [View] [Project Documentation]. The corresponding editor will then be started.

### 2.2.13 Printing a project

You can print all or just certain parts of the current project. The documents are printed on a standard printer configured under Windows. The print settings for your standard printer can be configured in the corresponding Windows dialog.



Before printing, you can use the [Extras] [FST Preferences...] menu command in the “Print” index to set the font and margin width for the printout (see also Fig. 2/16).

1. In order to print the current project or project part, select [Project] [Print...]. The “Print Project” window will then appear.

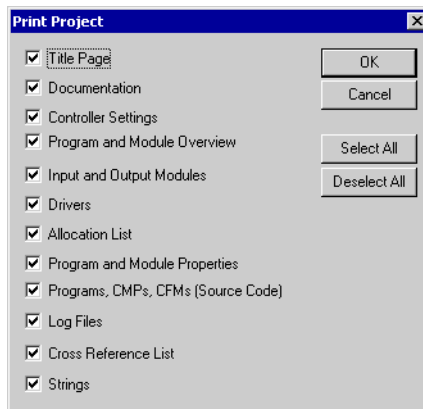


Fig. 2/15: Print Project

## 2. Basic functions of the FST software

Project parts	Description
Title Page	The title page contains the project name, information on FST version, date, time and the name of the user.
Documentation	The project documentation is printed only if the PROJECT.TXT file exists in the current project. If you have changed the name or extension of the file, you must print with the same editor you used to edit the file.
Controller Settings	Your set options.
Program and Module Overview	The program list shows all programs and modules in a sorted list with comment. The programs selected for downloading are tagged with a star.
Input and Output Modules	I/O configuration of the current project
Drivers	Driver configuration with settings for each driver
Allocation List	Allocation list for current project
Program and Module Properties	File size, date and other information on the program and/or module.
Programs, CMPs, CFMs (Source Code)	Source code for the program and modules
Log Files	The result of the last compilation for each program/module compiled in the project. The original rung for each imported program/module.
Cross Reference List	The cross reference list contains the operands used in the programs with lines/rungs for the occurrence in a sorted list. The database for creating the cross-reference list is created when the programs are compiled. The cross-reference list can therefore be printed only if the project has been compiled.
Strings	String list for the string driver

Tab. 2/5: Project parts

2. Select the project parts you wish to print and confirm with "OK". The printing operation will then start.

## 2. Basic functions of the FST software

A header and footer are printed on each page (apart from the title page). The header contains the project name, comment, controller type and name of the project part on the page. The footer contains the page number and print date.



The header and footer cannot be changed.

### Printing the content of the active program window

You can use the [Program] [Print...] command to print the content of the active program window.

### Print to file



To print to a file, we recommend you use the “Generic/Text Only” printer driver supplied with Windows (may need to be installed).

### Print format settings

You can set font and margins for the printout using the [Extras][FST Preferences...] command in the “Print” index.



Fig. 2/16: Print settings

If in the “Font” section you click on the “Browse” button, you must then select the printer. A list of supported fonts is then offered for selection. Click on “Default” to use the Windows default font.

You can use the “Print STL line numbers” checkbox in the bottom section of the window to prevent line numbers in STL programs from being printed out.



If you set narrower margins than permitted for the printer, the printout will be incomplete.

### 2.2.14 Archiving or restoring projects



Projects are archived as standard ZIP files. The ZIP format is a standard published in accordance with RFC 1950 to 1952 (Requests For Comments).

#### Archive project

Archive the current project as follows:

1. Select [Project] [Backup...].
2. In the standard Windows dialog that appears, select the directory and file name for the archive data and confirm your selection.

Once all restorable files have been deleted, the remaining project files are archived.

#### Restore project

To restore an archived project, proceed as follows:

1. Select [Project] [Restore...].
2. In the standard Windows dialog that appears, select the ZIP file that contains the required FST project.
3. In this window, enter a name for the project to be restored or select a project that you wish to overwrite. The project will then be restored and opened.

### 2.3 Replicating the hardware configuration

A compact or modular FEC/IPC system may consist of several cards and/or modules. Numerous additional functions may also occupy I/O address area. The address mapping is **not** determined by the physical set-up of the hardware. The start addresses for each occupied address area can be freely-assigned by copying the hardware configuration to FST. FST checks the configured address mapping for overlaps when the start addresses are entered.



#### Note

The hardware in a FST project must be correctly configured so that all cards, modules and functions used within the project occupy the correct input and output addresses.



The representation and options for hardware configuration with FST depend on the hardware in use. FST offers the most functions when configuring CPX terminals with CPX-FEC.

The hardware configuration is stored separately in the project for each controller type. If you change the controller type in the project, the configuration automatically changes to the corresponding type.

#### Opening the hardware configurator

Open configurator

Open the hardware configurator as follows:

 IO Configuration

- Select [View] [IO Configuration] or double-click on “IO Configuration” in the project window. The hardware configurator window will then open.

### 2.3.1 Hardware configuration for PS1, FEC Standard and FEC Compact

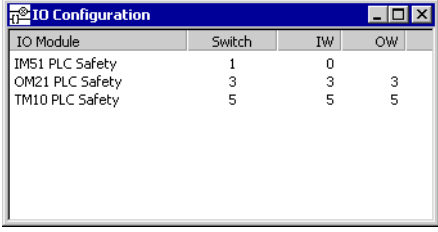
With PS1, FEC Standard and FEC Compact, the hardware configurator enables:

- the hardware configuration to be copied (tabulated display)
- free mapping of the start addresses to the respective occupied I/O address area
- the switch position for each I/O card to be set for clear identification. A specific I/O area of the process is assigned internally.



During configuration, FST checks the mapping of the internal I/O area and the address mapping for unacceptable overlaps.

For each inserted I/O card and/or function, the hardware configurator window displays name, switch position and start addresses for the assigned I/O address area.



IO Module	Switch	IW	OW
IM51 PLC Safety	1	0	
OM21 PLC Safety	3	3	3
TM10 PLC Safety	5	5	5

Fig. 2/17: Hardware configurator window  
(PS1, FEC Standard and FEC Compact)

The sequence of the items is determined by the entry sequence. During runtime, the cards are executed in the controller in the same sequence as they are listed in the configurator.



Notes on I/O configuration and addressing for PS1, FEC Standard and FEC Compact appear in Volume 2.



### Editing functions

The sequence of entries can be changed by cutting and pasting. The corresponding functions appear in the “Edit” menu or context menu. They always relate to the selected entries. Items are always pasted before the selection or, if nothing is selected, at the end of the list. The number of entries is displayed in the status line.



#### Note

Configuration changes cannot be cancelled using the [Edit] [Cancel] command. If the configuration contains a I/O card, the file for which does not exist in the FST library, the missing file is shown in brackets []. If the I/O configuration contains unknown cards, the project cannot be downloaded into the project.

During hardware configuration, objects may be inserted only if they exist for the configured controller type in the FST library. Where required, you can add any existing new objects to the FST library (see Section 2.10).

### Inserting IO Module

To insert a new entry, proceed as follows:

1. Select [Insert] [IO Module] or select [IO Module Entry] or double-click on an empty line on the list or press the INS key when the hardware configurator window is active. The following window will then open.

2. Basic functions of the FST software

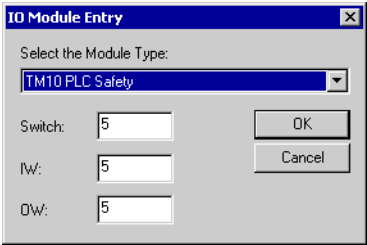


Fig. 2/18: Insert IO Module

Field	Description
Select the Module Type	Offers the I/O cards available for selection. If new hardware is available, you can expand the selection by adding I/O scripts using the runtime library (see Section 2.10). With some modules, there are several I/O scripts available. Select the suitable I/O script. Please note that some of the properties are determined by the selected I/O script. Others are determined by hardware preferences (e.g. via Jumpers) or via a special version of the card. Details appear in the documentation for the individual modules.
Switch	Switch position on the back of the card. Each I/O card occupies certain addresses in the internal I/O area of the process. Most I/O cards can operate in several areas. With these cards, the area can be selected by a rotary switch on the back. With some I/O drivers, the switch position is used for other purposes.
IW	Start addresses of the assigned input address area. Enter a start address from which you expect input data from the card in your programs. Some cards occupy more than one input word. These then also use subsequent words. The number of input words used by a card can be found in the corresponding card documentation.
OW	Start addresses of the assigned output address area. Enter a start address from which you expect the output data of the card to be addressed in your programs. Some cards occupy more than one output word. These then also use subsequent words. The number of output words used by a card can be found in the corresponding card documentation.

Tab. 2/6: Fields in the “IO Module Entry” window

## 2. Basic functions of the FST software

2. Make the required preferences and confirm with “OK”. A corresponding entry is then inserted into the list.

### Deleting an entry

Delete a selected entry as follows:

- Press the DEL key or select [Edit] [Delete] or select the [Delete] command from the context menu.

### Changing entry

Change a selected entry as follows:

- Select [View] [Properties] or the [Properties] command from the context menu or simply double-click on the corresponding entry. The “IO Module Entry” window in which you can make changes then opens.

### Printing

Print the current I/O configuration as follows:

- Select [Program] [Print...].

### Information on I/O cards for PS1

Most cards can operate in several I/O areas. With these cards, the area can be selected by a rotary switch on the back.

The documentation on the I/O cards describes the address mapping in the I/O area. However, some I/O modules require other preferences outside the IO area. Some of these options require preferences in the hardware (e.g. via Jumpers), while other properties can be programmed via software (e.g. FST I/O driver). The name of the I/O driver usually indicates these setting options.



Further details appear in the documentation on the I/O card. This documentation also indicates how many input and output words the card occupies. If a pure output card requires input words, they are usually used for status information.

### 2.3.2 Hardware configuration for CPX terminal (CPX-FEC)

The hardware configurator for the CPX terminal offers both offline and online functions. These include:

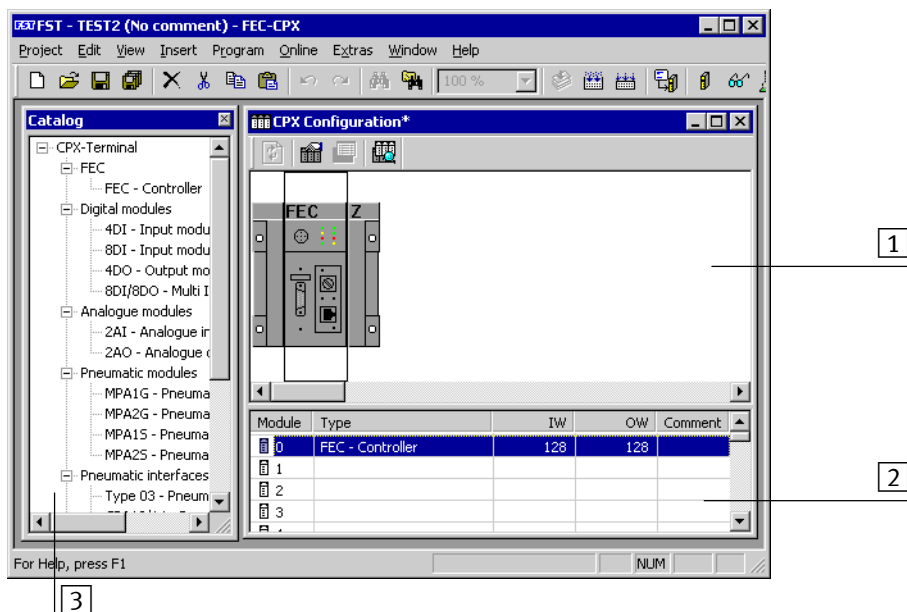
- graphic illustration of actual and nominal configuration of the CPX terminal
- upload of actual configuration
- generation of nominal configuration
- free mapping of the start addresses to the respective assigned I/O address area
- nominal/actual comparison of the hardware configuration
- definition of start of parameterising
- parameterising in online mode
- access to diagnostic information.



Use of the online functions requires a communication link between computer and controller (see also Section 2.9).

The first time the hardware configurator for CPX is started (see Section 2.3), a CPX-FEC is automatically present as Module 0 (see Fig. 2/29).

## 2. Basic functions of the FST software



- 1 Configuration illustration (graphic)      3 Catalogue
- 2 Configuration table

Fig. 2/19: Hardware configurator of the CPX terminal

Window/area	Description
Configuration illustration (graphic)	Graphic display of the hardware configuration.
Configuration table	Tabulated display of the hardware configuration. For each module, module position, type and start addresses of the occupied I/O address area are displayed in columns.
Catalogue	Offers the CPX modules available for selection.

Tab. 2/7: Windows and areas of the hardware configurator for the CPX terminal

2. Basic functions of the FST software

2.3.3 Online and offline mode of the hardware configurator (CPX-FEC)

The Online (online mode) and Editor (offline mode) can be used to switchover the hardware configurator accordingly. The commands are located:

- in the context menu of the hardware configurator window
- in the online menu, when the hardware configurator window is the active window.

The commands in the context menu of the hardware configurator window are automatically modified.

Context menu of the hardware configurator	
Offline (Editor active)	Online (Online mode active)
<div><div>System Settings</div><div>Default Settings...</div><div>Check Configuration</div><div>Actual-Nominal-Comparison</div><div>• Editor</div><div>Online</div><div>✓ Catalog</div><div>Properties</div></div>	<div><div>Refresh</div><div>SaveCtrl+S</div><div>System Settings</div><div>Diagnosis-Trace</div><div>Editor</div><div>• Online</div><div>Properties</div></div>

Fig. 2/20: Context menu of the hardware configurator

[Editor] (Offline mode) A dot in front of the [Editor] command indicates “Editor active” (Offline mode). All context menu commands relate to the illustrated nominal configuration.

[Online] A dot in front of the [Online] command indicates “Online mode active”. All commands relate to the actual configuration shown and/or the connected CPX terminal.

## 2. Basic functions of the FST software

### 2.3.4 Save the actual configuration as the nominal configuration (CPX-FEX)

Saving actual configuration as nominal configuration

If you have connected a CPX terminal with FEC to your PC, you can upload its configuration, address mapping and parameterisation and use as the nominal configuration.

Proceed as follows:

1. In the context menu of the CPX configuration window, select the [Online] command. The configurator then switches to online mode. The configuration and parameters of the connected CPX terminal are uploaded and displayed.

- 1 Active online connection is displayed in the title line

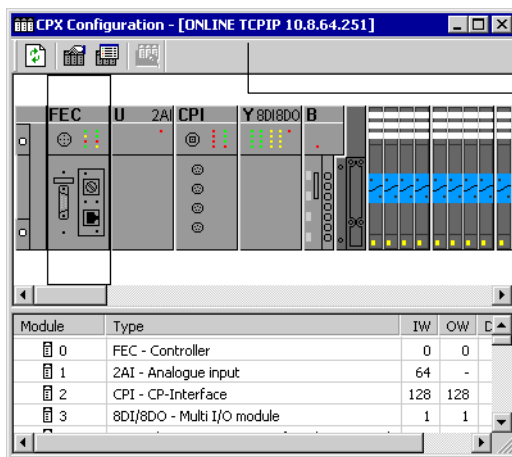


Fig. 2/21: Online mode of the hardware configurator (CPX terminal)

2. Now select the [Save] command from the context menu in the CPX configuration window. A prompt will then appear.
3. Confirm the prompt. The actual configuration will then be saved as the nominal configuration.

## 2. Basic functions of the FST software

4. To return to editor mode, select the [Editor] command from the context menu of the CPX configuration window.

You can use either all or any combination of actual configuration data as the nominal configuration. Proceed here as described in Section 2.3.5 below.



### 2.3.5 Performing nominal/actual comparison (CPX-FEC)

Actual nominal  
comparison

If required, you may compare the nominal and actual configurations. You can select which data are to be used as the nominal configuration:



1. In the context menu of the CPX configuration window, select the [Actual-Nominal-Comparison] command. The actual configuration of the connected CPX terminal is then uploaded and the “Actual-Nominal-Comparison” dialog window displayed. All differences between nominal and actual configurations are displayed.

- 1 Module position with checkbox column
- 2 Nominal configuration
- 3 Actual configuration
- 4 Information on the differences
- 5 Button for calling-up further information on the differences

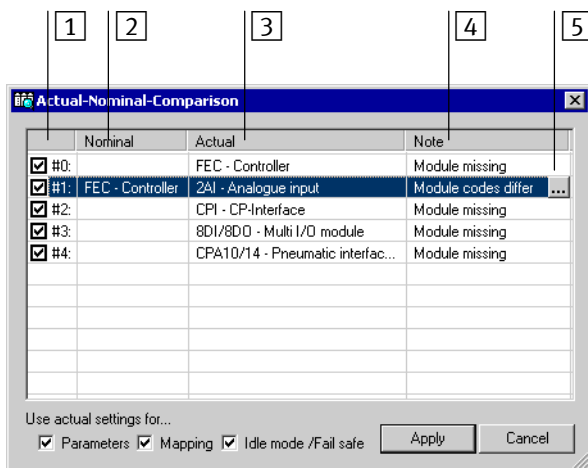
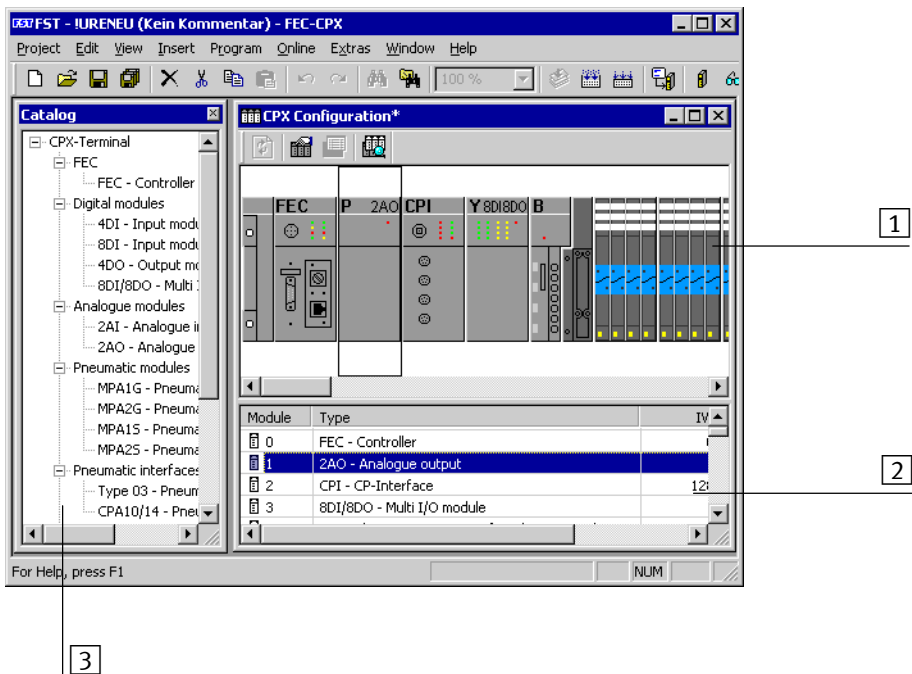


Fig. 2/22: Actual nominal comparison

- Select the actual configuration module you wish to use by ticking the checkbox in the module position column.
- If you wish to receive further information on the corresponding differences, click on button [5] in the “Note” column.

## 2. Basic functions of the FST software

- In the bottom section, select from “Use actual settings for...” either current parameter preferences, address Mapping or Idle mode/Fail safe settings.
2. To apply the selected configuration data, click on the “Apply” button. The selected data is then used as the nominal configuration and displayed.




- 1 Configuration illustration (graphic)
- 2 Configuration table
- 3 Catalogue

Fig. 2/23: Hardware configurator following application of actual configuration (example)

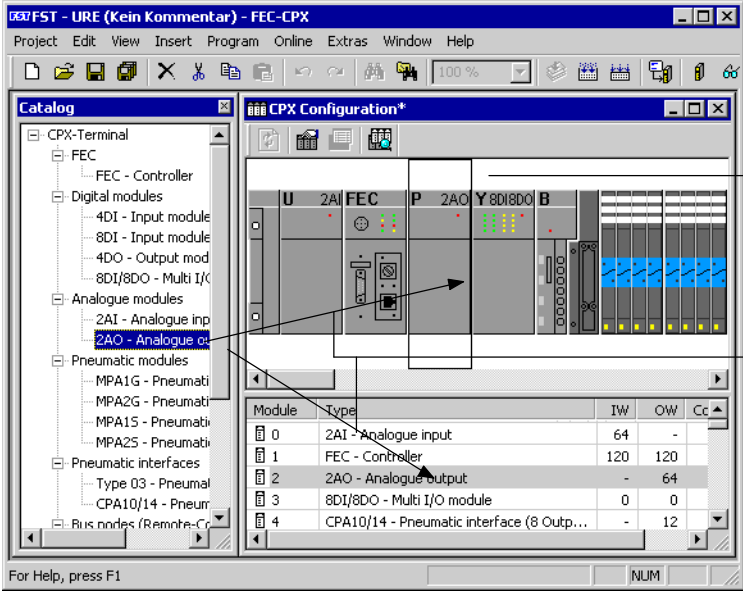
2. Basic functions of the FST software

2.3.6 Manually generating the nominal configuration (CPX-FEC)



**Note**  
Configuration changes cannot be cancelled using the [Edit] [Undo] command.

The nominal configuration can be conveniently generated using the drag & drop feature. You can insert and move modules in the graphic and tabulated displays using the drag & drop feature.



Module	Type	IW	OW	Cc
0	2AI - Analogue input	64	-	
1	FEC - Controller	120	120	
2	2AO - Analogue output	-	64	
3	8DI/8DO - Multi I/O module	0	0	
4	CPA10/14 - Pneumatic interface (8 Outp...	-	12	

1

2

1

2

1 Selected column      2 Insert by drag & drop

Fig. 2/24: Manually generating the nominal configuration (example)

Arrange the modules from left to right in line with the physical sequence on your CPX terminal.



Information on the various modules can be found in the CPX description for the corresponding module. An overview contains the description on CPX-FEC.

### Inserting and changing modules

The sequence of entries can be changed by drag & drop or by cutting and pasting. To do this, the “Catalog” window (see also Fig. 2/23) must be open.

#### Catalog

If the catalogue is closed, you can open it using the [View] [Catalog] command or the [Catalog] command in the context menu.

#### Inserting module

Insert a module as follows:

- Drag the module from the catalogue to the required position in the configuration table or the graphic display.

With pneumatic type 03 (Midi/Maxi) and type 12 (CPA), you cannot change the number of pneumatic modules using drag & drop. For these types, the number of occupied outputs is set on the pneumatic interface with a DIP switch (see also description for CPX I/O modules, P.BE-CPX-I/O..).

## 2. Basic functions of the FST software

### Changing number of pneumatic modules

To change the number of pneumatic modules for type 03 (Midi/Maxi) and Type 12 (CPA), proceed as follows:

1. In the CPX configuration window, select the relevant pneumatic interface.
2. Select [View] [Properties] or [Properties] from the context menu. The following window will then appear:

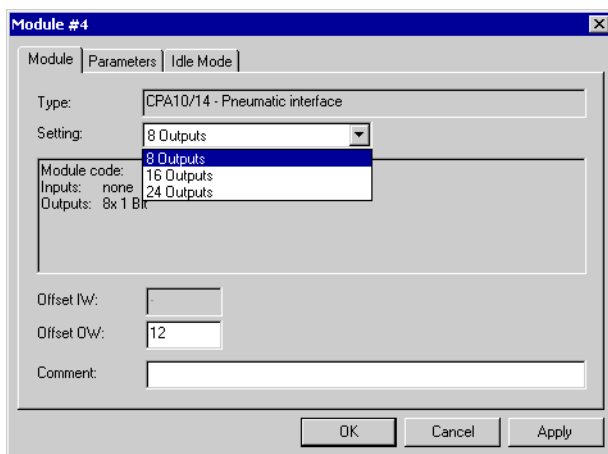


Fig. 2/25: “Module” dialog

3. In the “Module” index in the field “Setting”, set the desired number of output addresses to be occupied (consistent with DIL switch position at the pneumatic interface) and confirm with “OK”.



If the number of installed valve solenoid coils is lower than the number of the output addresses assigned by DIL switch, the surplus addresses are reserved for subsequent upgrades.

## 2. Basic functions of the FST software

### Deleting module

Delete a selected CPX module as follows:

- Press the DEL key or select [Edit] [Delete].

### Checking the configuration

A successful configuration is typified by:

- continuous occupation by modules
- clear mapping of the master.

### Checking configuration

### Checking configuration

You can check whether the generated configuration is permissible:

- Select the [Check Configuration] command from the context menu of the CPX configuration window. Then check whether the configuration is in principle permitted. You will then see information on how to remedy the error or the message:

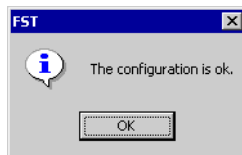


Fig. 2/26: Configuration check message (example)

Reset configuration

All parameters and the address mapping can be restored to the factory settings.

Resetting configuration      Restore the desired settings to factory settings as follows:

- 1. Select the [Default Settings] command from the context menu of the CPX configuration window. The following window will then appear:

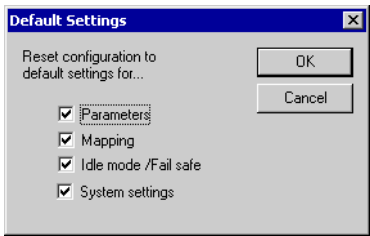


Fig. 2/27: Configuration check message (example)

- 2. Select the settings you wish to reset and confirm with OK.

Options	Description
Parameters	Module parameters without Idle mode and Fail safe
Mapping	Address mapping of the CPX terminal
Idle mode/Fail safe	Idle mode and Fail safe parameters
System settings	System parameters and memory parameter diagnosis

Tab. 2/8: Options – Reset configuration

## 2. Basic functions of the FST software

### 2.3.7 Setting and changing the address mapping (CPX-FEC)

Addresses are assigned automatically (auto mapping) in accordance with the following specification and may be changed as and when required.

Module type	Input word	Output word
Digital inputs and outputs	0 ... 31	0 ... 31
Valves	–	32 ... 63
Analog inputs and outputs	64 ... 127	64 ... 127
Technology modules	128 ... 255	128 ... 255

Tab. 2/9: Default address mapping

#### Changing address mapping

Change the address mapping as follows:

- Click on the address in the configuration table and change the value.



FST checks the assigned address mapping for unacceptable overlaps in the address area.

### 2.3.8 Parameterising the CPX terminal with FST (CPX-FEC)

The system reaction of the CPX terminal can usually be adapted to the relevant application through parameterisation. A distinction is made between the following parameterisations:

- System parameterising, e.g.: switching out fault messages, setting reaction times, etc.
- Module parameterising (module and channel-specific), e.g.: monitoring, settings in the event of faults, settings for forcing.
- Parameterising the diagnostic memory.



## 2. Basic functions of the FST software

The CPX terminal also provides system data, module data and diagnostic data.



A detailed description of the individual parameters and data, as well as basic information on parameterisation be found in the CPX system manual (P.BE-CPX-SYS-..).



The module parameters which are available for the various modules can be found in the manual for the corresponding module (e.g. manual for CPX pneumatic interfaces and CPX I/O modules (P.BE-CPX-EA-..)).

The parameters and data of the CPX terminal are grouped in FST as follows:

Group	Description
System settings	<ul style="list-style-type: none"><li>– System data</li><li>– System parameters</li><li>– Diagnostic memory parameters</li><li>– Group diagnosis</li></ul>
Properties	<ul style="list-style-type: none"><li>– Module data</li><li>– Module parameters</li><li>– Module diagnostic data</li><li>– Channel-specific module parameters</li></ul>

Tab. 2/10: Grouping of parameters and data in FST

Parameterisation takes place as follows:

- the nominal configuration (offline); parameters are transferred by downloading the project.
- the actual configuration (online); parameters can be transferred immediately.

Once a dialog has been opened, you can view and change its parameters.

2. Basic functions of the FST software

System settings



Open the system settings as follows:

- In the context menu of the CPX configuration window, select the [System Settings] command.

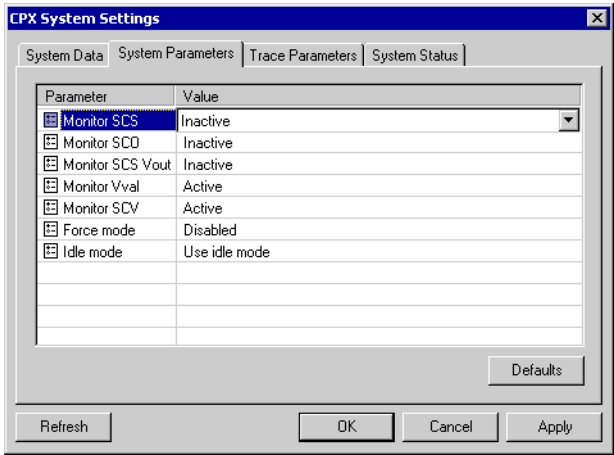


Fig. 2/28: CPX system settings (in this case, online mode)

Index	Description
System Data (online only)	Shows important CPX terminal system data and contains the “Clear Force table” button (immediately deletes all channel-specific force settings).
System Parameters	Enables system parameters to be changed and contains the “Defaults” button (resets the displayed settings to factory settings).
Trace parameters	Enables diagnostic memory parameters to be changed and contains the “Defaults” button (resets the displayed settings to factory settings).
System Status (online only)	Displays the group diagnostic message and contains the “Trace” button (displays the content of the diagnostic memory).

Tab. 2/11: Tabs for the system settings

## 2. Basic functions of the FST software

### Module properties

Open the module properties as follows:

- Select the relevant module and in the context menu of the CPX configuration window, select the [Properties] command.

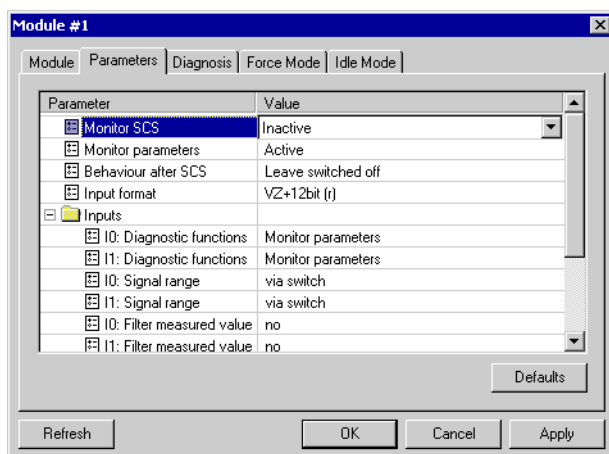


Fig. 2/29: Module parameters (in this case, online mode)

Index	Description
Module	Displays important module data and number of occupied input addresses.
Parameters	Enables module parameters to be changed and contains the “Defaults” button (resets the displayed settings to factory settings).
Diagnosis (online only)	Displays the module diagnostic data and contains the “Trace” button (displays the content of the diagnostic memory).
Force Mode (online only)	Enables channel-specific Force settings to be changed and contains the “Clear table” button (immediately deletes the displayed force settings).
Idle Mode	Enables Idle mode settings to be changed.

Tab. 2/12: Tabs for the module settings

## 2. Basic functions of the FST software

### Buttons for parameterising

Button	Offline	Online
[Defaults]	Resets all parameters of the tab to the factory setting.	Resets all parameters of the tab to the factory setting.
[Refresh] (online only)	–	Parameters are uploaded again and displayed.
[OK]	Parameter changes are applied to the current nominal configuration and the dialog is closed.	Parameter changes are used for the current nominal configuration and the dialog is closed.
[Cancel]	Cancels the operation and closes the dialog	
[Apply]	Parameter changes are applied to the current nominal configuration; the dialog remains open.	Parameter changes are transferred to the CPX terminal; dialog remains open.

Tab. 2/13: Buttons for parameterising

#### Changing parameters

Change the parameters as follows:

1. Online: Active online mode in the context menu.  
Offline: Active edit mode in the context menu.
2. Set the system parameters or diagnostic memory parameters: In the context menu, select the [System Settings] command.  
Setting module parameters: In the context menu, select the [Properties] command.
3. Select the tab for the parameter type (see also Tab. 2/11 and Tab. 2/12).
4. Change the desired parameters as follows.


## 2. Basic functions of the FST software



### Caution

Incorrect parameterising can cause physical injury and damage to property.

- When setting the parameters, always observe the instructions in the CPX system description and/or the description for each device or module.

As a general rule, select the desired setting for the parameter from the list field. Some parameters feature several settings. In these cases, open a dialog for parametrising by clicking the button  in the “Value” column (see Fig. 2/29).

5. In online mode:  
If you want to transfer the settings you have made, confirm with “Apply” or “OK”.  
The new parameter settings will then be transferred to the connected device.

In offline mode:

If you wish to apply the new settings to the nominal configuration, confirm with “Apply” or “OK”.



Refresh

### Uploading configuration and parameters again

In online mode, you can upload the configuration and parameters again using the [Refresh] command in the context menu.

2.3.9 Trace memory

The hardware configurator enables you to view the content of the diagnostic memory for your CPX terminal.



- When the configurator is in online mode, select the [Trace Memory] command from the context menu. The contents of the diagnostic memory will then be displayed.

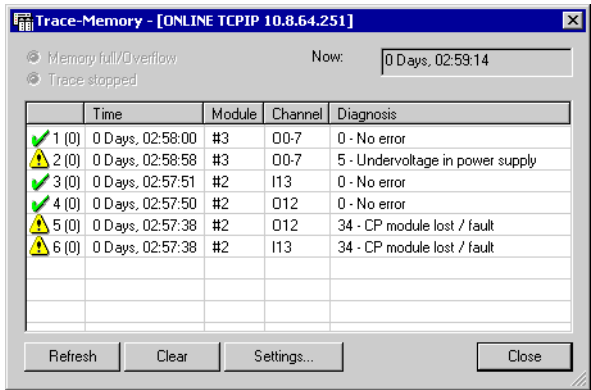


Fig. 2/30: Uploading trace memory in online mode using the CPX configurator

Button	Description
Refresh	Uploads trace memory again and displays
Clear	Deletes all entries from the trace memory
Settings...	Enables trace memory parameters to be changed

Tab. 2/14: Buttons in the “Trace Memory” dialog



Further information on the trace memory of the CPX terminal can be found in the CPX system manual.

### 2.4 The driver configuration

Certain functions in the controller are supported by drivers (e.g. fieldbus function). These drivers are usually activated when the controller is restarted before the main program runtime. FST supports the handling of these drivers. It manages the downloading operation and preparations for automatic start of the drivers. The drivers are selected and set for the project in the driver configurator.

Each driver has a number. This number is unique although drivers that serve the same purpose may have the same number. Only drivers with different numbers may be used within a project. Some drivers use the functions of other drivers. In this case, these drivers must also be configured in the project. Most drivers feature options that can be individually set.

Some drivers require a complex configuration, e.g. the fieldbus configuration. This configuration is usually created using a special configuration program and is saved to a file. The content of this file is integrated into the project file and thus downloaded to the controller where it is used by the driver.

The driver configuration is stored separately in the project for each controller type. If you change the controller type in the project, the configuration automatically changes to the corresponding type. Initially, the driver configuration for each controller type is empty.

Drivers may only be added to the driver configuration if they exist for the configured controller type in the FST library.



Detailed information on the various drivers and modules appears in Volume 2.

2. Basic functions of the FST software

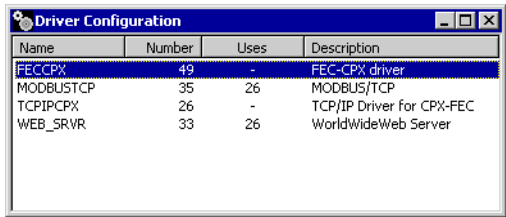
Opening the driver configurator

Opening the driver configurator



Open the driver configurator as follows:

- Select [View] [Driver Configuration] or double-click on “Driver Configuration” item in the project window. The driver configurator window will then open.

A screenshot of the 'Driver Configuration' window. It has a title bar with a gear icon and standard window controls. The window contains a table with four columns: Name, Number, Uses, and Description. The first row is highlighted in blue.

Name	Number	Uses	Description
FECCPX	49	-	FEC-CPX driver
MODBUSTCP	35	26	MODBUS/TCP
TCP/IPCPX	26	-	TCP/IP Driver for CPX-FEC
WEB_SRVR	33	26	WorldWideWeb Server

Fig. 2/31: Driver configurator (example)

The current driver configuration is displayed. For each entry, the name, number, any other required drivers and a brief description is displayed in columns.

2.4.1 Editing functions for driver configuration



Note

Configuration changes cannot be cancelled using the [Edit] [Undo] command. They are saved immediately!

Drivers may be added only if they exist for the configured controller type in the FST library. Where required, you can add any existing new drivers to the FST library (see Section 2.10).



## 2. Basic functions of the FST software



### Note

Any drivers contained in the configuration for which descriptions do not exist in the FST library are shown in brackets []. The settings for this driver are displayed (if present), although cannot be changed. If the driver configuration contains an unknown driver, the project cannot be downloaded.

### Inserting drivers

Insert a driver as follows:

1. Select [Insert] [Driver...] or select [Select Driver] from the context menu or double-click on an empty line on the list or press the INS key when the hardware configurator window is active.

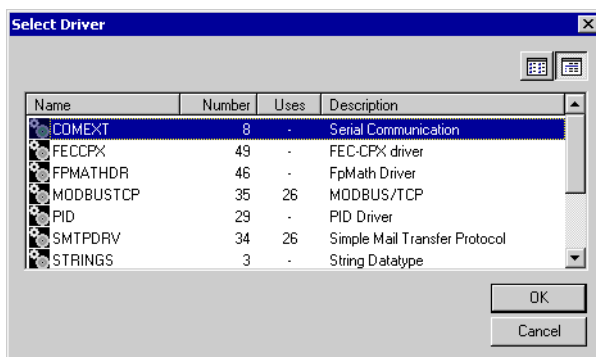


Fig. 2/32: Selecting drivers

2. Select the required driver and confirm with "OK". A dialog appears for making the driver settings.

## 2. Basic functions of the FST software

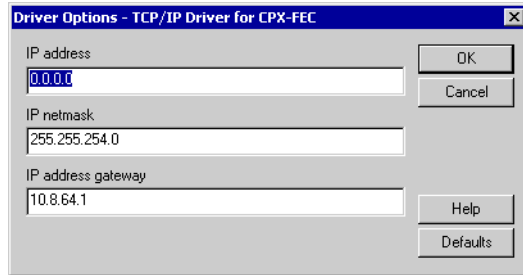


Fig. 2/33: Driver settings (example)

The required settings are specified in the supplied driver description file and are different for each driver. Default values exist for each setting. You can use the “Defaults” button to call-up these defaults for all settings.

Most drivers require a PLC drive. Enter the PLC drive to which the executable driver file is to be downloaded or on which it is located. Before the project is downloaded, check whether the corresponding drive exists in the control.

The “Help” button shows you how to use the drivers.

3. Make the required settings and confirm with “OK”. The driver is then added to the list.

### Viewing and changing driver properties

View and change the properties for the selected drivers as follows:

- Select [View] [Properties] or the [Properties] command from the context menu or simply double-click on the corresponding entry. The corresponding dialog will then open (see Fig. 2/33).

## 2. Basic functions of the FST software

### Deleting a driver

Delete a selected driver from the list as follows:

- Press the DEL key or select the [Edit] [Delete] menu command or select the [Delete] command from the context menu.

### Printing

Print the current driver configuration as follows:

- Select [Program] [Print...].

## 2.5 Controller settings (PLC settings)

You can save certain PLC settings in the project. These settings become effective once the project is downloaded.

Make the PLC settings as follows:

### Controller Settings

1. Double-click the PLC settings item in the project window or select the [View][Controller Settings] menu command. The PLC Settings dialog window will be displayed.
2. Make the required settings and to finish confirm with “OK”.



The various settings in the individual tabs are written to the following sections.

### 2.5.1 Settings for runtime behaviour

Make the settings for the runtime behaviour of your controller in the “Run Mode” tab of the “Controller Settings” dialog (see also Section 2.5).

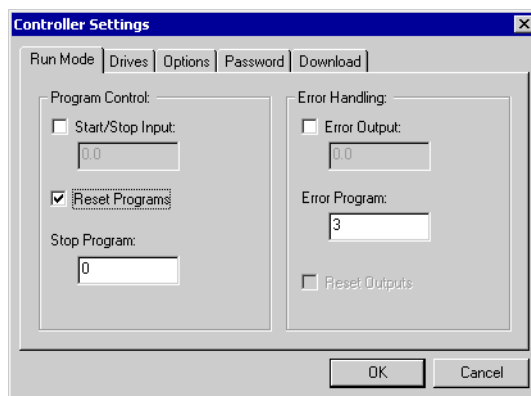


Fig. 2/34: “Run Mode” tab

## 2. Basic functions of the FST software

Field	Description	Default
Start/Stop Input	<p>Indicates whether an input is to be used as a start/stop input. If this is the case, enter the input to be used into the input field (see also Section 5.3).</p> <ul style="list-style-type: none"> <li>When you select a start/stop input, the “Autostart after download” function will automatically deactivate when selected (see also Tab. 2/20).</li> <li>The start/stop input is active only if the start/stop switch of the controller is set to “Start”. With Power ON, it is level-sensitive and edge-sensitive during operation (positive edge for start signal, negative edge for stop signal).</li> <li>It is not checked whether a physical input card is configured for this input. If this is not the case, the input always reports 0-signal unless it is explicitly set by a CI command or a driver.</li> </ul>	Inactive
Reset Programs	<p>Specifies whether on stop, all programs are to be reset or simply halted via the start/stop switch and/or whether on start, all halted programs are to be continued or only P0 is to be continued and/or restarted via the start/stop switch (see also Tab. 5/26).</p> <ul style="list-style-type: none"> <li>If an error occurs and an error program is not present, all programs are reset irrespective of the setting.</li> <li>Select this option if you do not want, as well as activating the P0 program, all programs to be stopped (halted). Reactivate programs when the start/stop switch is set to “Start” (rising edge).</li> <li>When a start signal is received, all halted programs are continued via the online control panel or the “R” CI command. Use the “RPO” CI command if you want to continue only the program 0.</li> </ul>	Active
Stop Program	<p>Specifies the program to be started via the start/stop switch or the online control panel when the system stops. The stop program enables a system to be moved to a safe operating status in the event of a stop. It is not started if the controller is stopped by an error (see info. under Error Program).</p> <p>Programs 1 to 63 are permissible. 0 denotes “No stop program”. Please note that a restart is possible when the stop program is still active, e.g. via a corresponding lock in the start program, such as:</p> <pre> STEP "" Wait until stop program 42 has finished WHEN      N      P42 THEN      NOP </pre>	0 (no stop program)

Tab. 2/15: Fields for “Run Mode” “Program Control” tab

## 2. Basic functions of the FST software

Field	Description	Default
Error Output	Indicates whether an output is to be set in the event of an error. If this is the case, enter the output to be used into the input field. <ul style="list-style-type: none"><li>– It is not checked whether a physical output card is configured for this output.</li><li>– As long as the error is still present, the error output is not reset.</li></ul>	Inactive
Error Program	Specifies the program to be started in the event of an error (see also Section 5.5.2). Programs 1 to 63 are permissible. 0 denotes “No error program”.	0 (no error program)
Reset Outputs <sup>1)</sup>	Only for PS1, FEC Standard and FEC Compact <sup>1)</sup> : Specifies whether all outputs are to be reset when: <ul style="list-style-type: none"><li>– the start/stop switch is set to “Stop” (negative edge)</li><li>– the “S” CI command is executed</li><li>– an error occurs and no error program is present</li></ul>	Active
<sup>1)</sup> With the CPX terminal, this setting is set via the idle mode parameter.		

Tab. 2/16: Fields for “Run Mode” “Program Control” tab

### 2.5.2 Drives

Make the settings for the drives of your controller in the “Drives” tab of the “Controller Settings” dialog (see also Section 2.5).

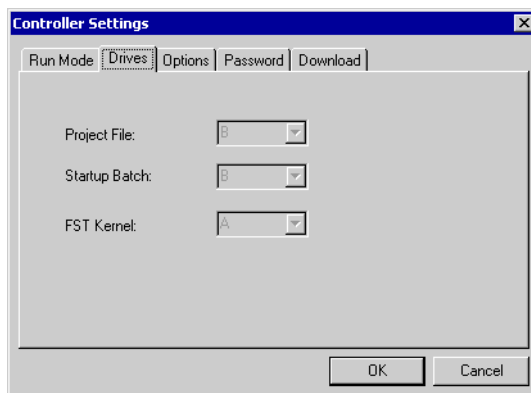


Fig. 2/35: “Drives” tab

## 2. Basic functions of the FST software

A drive may be changed only if it is supported by the controller type in use.

With PS1: If the required drive is not listed in the selection box, simply enter the respective letter codes. Before downloading, check whether the entered drive exists in the controller.

Field	Description	Default
Project File	Drive on which the project file is to be stored in the controller.	CPX-FEC: B FEC Compact: B FEC Standard: B
Startup Batch	Drive on which the "STARTUP.BAT" start batch file is to be stored in the controller. This is executed automatically on start.	HC0X: B HC1X: C HC2X: C
FST Kernel	Drive on which the FST main program runtime is to be stored in the controller.	CPX-FEC: A FEC Compact: A FEC Standard: A HC0X: A HC1X: A HC2X: C

Tab. 2/17: Fields of the "Drives" tab

2. Basic functions of the FST software

2.5.3 Options

Make the settings for the options of your controller in the “Options” tab of the “Controller Settings” dialog (see also Section 2.5).

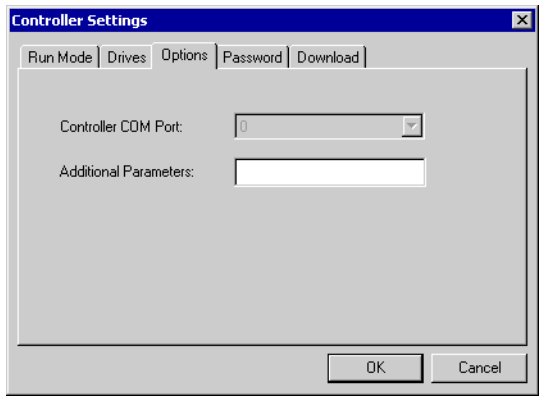


Fig. 2/36: “Options” tab

Field	Description	Default
Controller COM Port	Number of the serial port on the controller that is to be used by the FST PLC operating system for CI communication.	CPX-FEC: 0 FEC Compact: 0 FEC Standard: 0 HC0X: 0 HC1X, HC2X: 1
Additional Parameters <sup>1)</sup>	Additional parameters for the main program runtime. From main program runtime Version 2.26 or higher: – The -v<baud> parameter can be used to define the baudrate for the above PLC COM interface. Here, <baud> denotes the baudrate value (1200, 2400, 4800, 9600, 19200, 38400, 57600). In the absence of this parameter (empty input field), the standard baudrate applies. Example: -v1200 denotes 1200 baud; -v38 denotes 38400 baud <sup>3)</sup>	. <sup>2)</sup>
<sup>1)</sup> The baudrate can be changed by CI command “MV” (see Chapter 5.6.6) or via a hardware BREAK. <sup>2)</sup> Depends on the controller. <sup>3)</sup> The value for the baudrate can be abbreviated to the first 2 characters. -v56 and -v57 denote 57600 baud.		

Tab. 2/18: Fields of the “Options” tab



## 2. Basic functions of the FST software

### 2.5.4 Password protection

Data, operands and states in the controller can be password protected to prevent accidental changes. When password protection is active, there is no modify access and no driver-specific CI commands can be selected.



#### Note

Do not forget the password! The password is generally saved in the connected device. Password protection remains active even when a dialog device is active. It is not, however, connection-oriented. Password protection can be activated or deactivated equally by all communication partners.

The password protection does not offer permanent protection against malicious manipulation, but only protection against unintentional modification by unauthorized third parties. There is no repetition limit for password entry. You can define a password as follows:

Type of password allocation	Description
Online password allocation: <ul style="list-style-type: none"><li>– with the online control panel (see section 2.9.2)</li><li>– by CI command (see section 5.6.10)</li></ul>	Password can be permanently changed or deleted via the online control panel or the CI command. <sup>1)</sup> Password protection is activated: <ul style="list-style-type: none"><li>• when the project is restarted (e.g. Power OFF/ON). <sup>1)</sup></li><li>• via the online control panel or CI command.</li></ul>
Store password in the project: <ul style="list-style-type: none"><li>– in the “Password” tab with the Controller settings (see Fig. 2/37)</li></ul>	Password is transferred and permanently stored in the project on downloading. Changes via the online control panel or CI command therefore have only a temporary effect. Password protection is activated: <ul style="list-style-type: none"><li>• when the project is restarted (e.g. Power OFF/ON).</li><li>• via the online control panel or CI command.</li></ul>
<sup>1)</sup> The password is then only stored residually if operands are also stored residually (see also Section 5.1.1).	

Tab. 2/19: Type of password allocation



Recommendation: Save the password in the project only if your controller provides no retentive memory for operands and the password (see also Section 5.1.1). Use the option of entering the password online. This makes it easier to permanently change or delete the password.

### Storing the password in the project

- To store the password in the project, enter the desired password into the “Password” tab of the “Controller Settings” dialog.

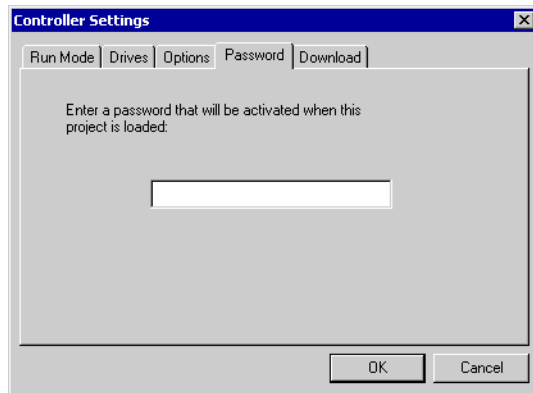


Fig. 2/37: “Password” tab

A password consists of between 3 and 20 visible ASCII characters. Separators such as commas, spaces, tab, IBM extended characters etc. are not permitted.

If you do not wish to permanently store a password in the project, leave the input field empty.

## 2. Basic functions of the FST software

### 2.5.5 Settings for downloading

Make the settings for downloading in the “Download” tab of the “Controller Settings” dialog (see also Section 2.5).

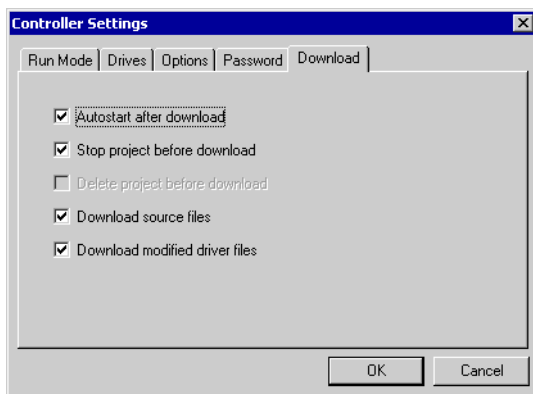


Fig. 2/38: “Download” tab

Field	Description	Default
Autostart after download	Specifies whether the project is to be started automatically by FST following download. <ul style="list-style-type: none"><li>– This setting can be activated only if no start/stop input has been specified (see Tab. 2/15).</li><li>– This setting is effective only if no password has been stored in the project (see Fig. 2/37).</li></ul>	Inactive
Stop project before download	Specifies whether an active project in the controller is to be halted before download.	Inactive
Delete project before download	Specifies whether an active project in the controller is to be removed from the working memory before download. If this option is active, the project will continue to run uninterrupted in the controller while the new project is being downloaded. <ul style="list-style-type: none"><li>– Please note that this option is ignored if you wish to download via TCP/IP, as the TCP/IP driver would otherwise be stopped.</li></ul>	Inactive

## 2. Basic functions of the FST software

Field	Description	Default
Download source files	Specifies whether all sources required to restore the project are to be stored in a ZIP file in the controller. The downloaded sources can be restored using the [Online][Upload Project] menu command.	Inactive
Download modified driver files	<p>Specifies whether the driver files are to be transferred to the control.</p> <ul style="list-style-type: none"><li>• Deactivate this function if you want to avoid overwriting driver files already present in the controller, if, for instance, they were downloaded using a different FST version.</li></ul> <p>These settings do not affect their respective settings as the configuration data are stored in the project file.</p>	Active

Tab. 2/20: Fields of the “Download” tab

### 2.6 The allocation list

The operating resources for the controller (timer, counter, index, inputs, outputs etc.) are addressed via operands (see also Section 5.1). A distinction is drawn here between single-bit and multi-bit operands. Inputs, outputs and flags are organised by word. They can be addressed either by bit or word. All other operands can be addressed by only bit or only word. For each program, the operating resources can be accessed via:

- symbolic operands (e.g. MOTOR for output 1.0)
- absolute operands (e.g. T1 for Timer No. 1).

#### Symbolic operands

You can specify any symbolic name for the operand. If, for instance, output O5.3 switches a motor on and off, you can define this output in your programs as “MotorOn”. Symbolic operands can be assigned to absolute operands in the allocation list.

Symbolic operands help you make your programs more transparent and ensure clarity, even when operands occur in large numbers.



**Recommendation:** Generate the allocation list before entering the program and use only symbolic operands in the programs. It is also possible to insert operands in the allocation list while the program is being entered.

The name must not contain more than 9 characters and must start with a letter or an underscore (\_). Any combination of letters, numbers and the underscore can then follow; spaces are not permitted. A valid, absolute operand is not permitted as a symbolic operand.

## 2. Basic functions of the FST software

### Absolute operands

An absolute operand consists of:

- a character denoting the operand type (O for output, I for input etc.)
- where applicable the character W to denote word (for multi-bit operands)
- the address of the operating resources.

### Multi-bit operands

An absolute multi-bit operand contains the following components:

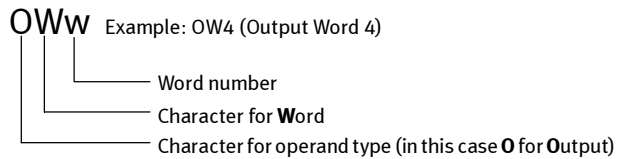


Fig. 2/39: Absolute multi-bit operand

### Single-bit operands

An absolute single-bit operand contains the following components:

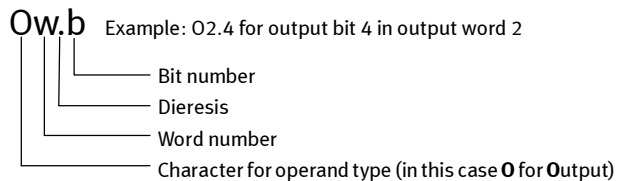


Fig. 2/40: Absolute single-bit operand

### Example

The second output word is designated OW2, for example.  
The output 4 in output word 2 is designated O2.4.

## 2. Basic functions of the FST software

### Opening the allocation list editor

Opening the allocation list editor

Open the allocation list editor as follows:



- Select [View] [Allocation List] or double-click on the “Allocation List” item in the project window. The allocation list editor will then open.

Operand	Symbol	Comment
00.0	Rel_open	Relay: Start motor to open the door
00.1	Rel_close	Relay: Start motor to close the door
I0.0	open	Limit switch (open)
I0.1	closed	Limit switch (closed)
I0.2	Inside_Op	Switch inside: Open the door
I0.3	Inside_cl	Switch inside: Close the door
I0.4	Outs_open	Switch outside: Open the door
I0.5	Outs_clos	Switch outside: Close the door
I0.6	Key_switc	Key switch outside

Fig. 2/41: Allocation list editor (example)

The current allocation list is displayed. For each item, the absolute operand, the symbolic operand and a comment is displayed in columns. The symbols at the start of the line help to differentiate between the inputs and outputs of internal operands.

## 2. Basic functions of the FST software

### 2.6.1 Functions of the allocation list editor



#### Note

Changes to the allocation list cannot be cancelled using the [Edit] [Undo] command. They are saved immediately!

#### Editing regulations

Observe the following regulations when editing:

- An absolute or symbolic operand may only appear in the allocation list once.
- A valid, absolute operand may not be used as a symbolic identifier.
- An absolute operand can be entered with or without symbolic identifier. It is not permitted to enter a symbolic identifier without an absolute operand.

#### Inserting an operand

Insert an operand as follows:

1. Select [Insert] [Operand...] or select [Insert operand] from the context menu or double-click on an empty line on the list or press the INS key when the allocation list window is active.

The screenshot shows a dialog box titled "Allocation List Entry" with a standard Windows-style title bar (blue with a close button). Inside the dialog, there are three text input fields stacked vertically. The first is labeled "Absolute Operand:", the second "Symbolic Operand:", and the third "Comment:". To the right of these fields are two buttons: "OK" and "Cancel".

Fig. 2/42: Allocation List Entry



## 2. Basic functions of the FST software

Field	Description
Absolute Operand	Type and address of the operand (e.g. O2.4 or FW1 etc.)
Symbolic Operand	Any symbolic designation of the operand must not consist of more than 9 characters (e.g. MotorOn). The first character must be a letter or an underscore (_). Spaces and the following characters are not permitted: \ / : * ? . < >  .
Comment	Brief explanation on the operand; must not contain more than 36 characters.

Tab. 2/21: Input fields of the “Allocation List Entry” dialog

2. Enter the desired absolute operand. If required, you can also enter a symbolic operand and a comment.
3. Confirm with “OK”. The entry will then be accepted into the allocation list.

If you use an operand from the allocation list in a program, the corresponding allocation list comment is automatically inserted into the program text.



New items can be entered into the allocation list even when a program is being entered (see Sections 3.2.2 and 4.2.3).

### Changing allocation list entry

Change a selected allocation list entry as follows:

- Select [View] [Properties] or the [Properties] command from the context menu or simply double-click on the corresponding entry. The corresponding dialog will then open (see Fig. 2/42).



You can use the [Find...] command in the [Edit] menu to search for text in the allocation list.

## 2. Basic functions of the FST software

### Deleting an allocation list entry

Delete a selected entry from the list as follows:

- Press the DEL key or select [Edit] [Delete] or select the [Delete] command from the context menu.

### Printing

Print the current allocation list as follows:

- Select [Program] [Print].



You can also create and edit the allocation list using a table editor, e.g. Excel. Select the area to be copied and/or insert location and use the cut, copy and paste functions in the [Edit] menu, either in FST or the table editor program.

### 2.7 The Strings Editor

You can use the Strings Editor to edit the initialisation strings for the STRINGS driver.



Detailed information on use of the STRINGS driver appears in Volume 2.

Strings are initialised using a simple text file. These can be conveniently edited using the installed Strings Editor.

Opening the Strings Editor    Open the Strings Editor as follows:



- Select [View][Strings] or double-click on the “Strings” item in the project window. This starts the Editor and the current initialisation strings will then be displayed.

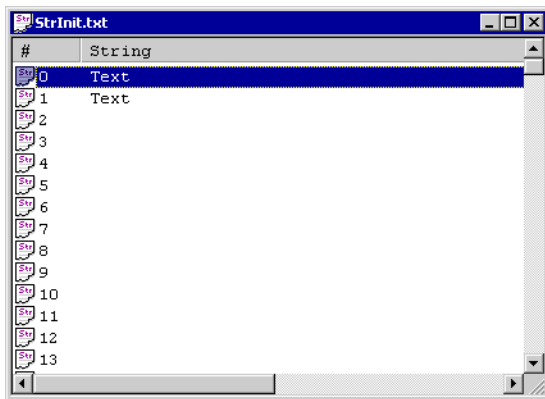


Fig. 2/43: Strings Editor

Each initialisation string is displayed together with its number. The number of characters currently being used is displayed in the status line.

## 2. Basic functions of the FST software

### 2.7.1 Functions of the Strings Editor

Use the clipboard functions of the [Edit] menu to copy or transfer strings to or from another application. You can also move and/or copy strings within the list using the drag & drop feature. To copy, keep the CTRL key pressed down.

Select [Program] [Print] to print the current strings. The corresponding functions in the [Edit] menu enable you to search and replace strings.



#### Note

All strings from the initialisation file are shown, even when there are more than were entered into the STRINGS driver.

If you change the file name for the strings in the settings for the STRINGS driver, the entries from the old file are **not** automatically transferred. If you want to transfer the strings, use the clipboard functions in the “Edit” menu.

#### Deleting strings

Delete strings as follows:

- Selected strings are deleted simply by pressing the DEL key or selecting [Edit] [Delete] from the menu.

#### Inserting strings

Insert strings as follows:

- Strings are inserted before the current selection by pressing the INS key. If you want to insert a new item at the end, make sure that no item is selected.

## 2. Basic functions of the FST software

### Changing strings

Change strings as follows:

- If you want to change a string, click on it or select it and press the ENTER key.
- Click outside the entry field or press ENTER again to accept the changed string.
- Press ESC to restore the original value.

### Special characters

Special characters are shown by combining two characters, the '\ ' and one other. The following special characters are possible:

Special characters	Meaning	Description
\a	alert	Bell symbol (beep)
\b	backspace	Positioning by one character back
\f	formfeed	Form feed (FF)
\n	linefeed	Line feed (LF)
\r	return	New line (CR)
\t	tab	Tab character
\<No.>		Hexadecimal definition of a character; <No> must start with a number, e.g. "\008" (correct) instead of "\08" (incorrect).
\\	\	The character '\ ' is shown by two '\\ '.

Tab. 2/22: Special characters

### 2.8 Managing the controller programs

Programs and function and program modules are executable parts of the project. A project can comprise as a maximum:

- 64 programs
- 100 function modules (CFMs)
- 100 program modules (CMPs).



As they are managed in a similar manner, they are all described below as programs.

FST manages up to 9 different versions of each program. Only one of these versions can be downloaded into the controller.

FST can create programs in statement list (STL) or ladder diagram (LDR). For special tasks, you can use the pre-compiled modules in the FST library. These are mainly programmed in C.

### 2.8.1 Creating a new program or module



#### Note

Observe the following regulations when creating programs or modules.

- Only program 0 is automatically activated at system start.
- If you create a program that already exists, the old one is overwritten following a prompt.
- If you create a program with the same number but a higher version number, a copy of the previous version is automatically transferred providing the programming language is consistent.
- CI commands RB and RF use the local operands of program P63. You should therefore avoid using this program number if you want to call modules via CI command.

Create a new project or a new module as follows:



1. Select [Program] [New...] or the [New...] command in the context menu of the project window via the corresponding element. The “New Program” dialog will then open.

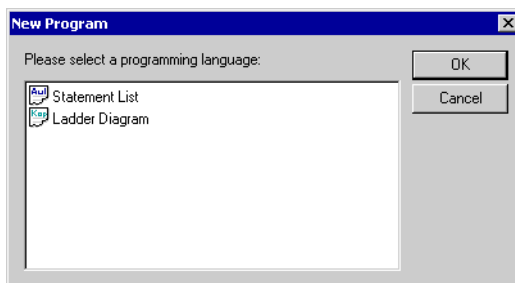


Fig. 2/44: Selecting programming language

2. Basic functions of the FST software

2. Select the programming language. The programs and/or modules present in the project are then displayed with comment:

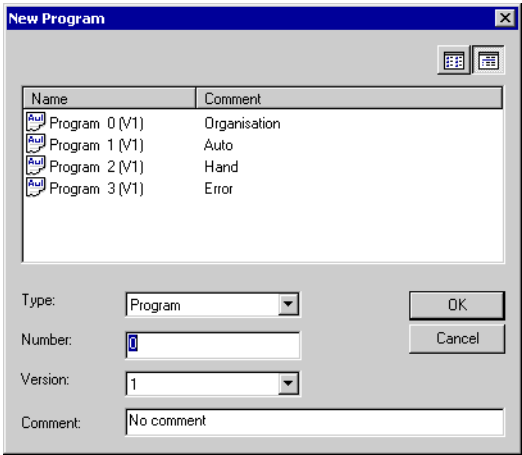


Fig. 2/45: Select program properties

Field	Description
Type	<ul style="list-style-type: none"> <li>– Program</li> <li>– CMP stands for program modules (sub-routine with or without steps)</li> <li>– CFM stands for function modules (sub-routine without steps)</li> </ul>
Number	Identifies the program or the module. Permitted range: Programs: 0 to 63 *) CMPs: 0 to 99 CFMs 0 to 99 Program and module numbers may only be assigned once per project.
Version	FST can manage up to 9 different versions of a program. Only one version can be downloaded.
Comment	To aid program identification, you can enter descriptive text on the program here (max. 255 characters).
*) Only program 0 is automatically activated at system start.	

Tab. 2/23: Program properties



## 2. Basic functions of the FST software

3. Make the required entries and confirm with “OK”. The new program and/or module is then created and automatically opened for editing.



The new program and/or new module is entered into the project window at the relevant location, from which it can be directly selected. A tick before the entry indicates that the new object has been selected for download (see also Section 2.8.8). It is transferred as the project is downloaded into the controller.

### 2.8.2 Importing programs and modules

The import function of FST enables you to:

- Import modules from the FST module library into the current project,
- Import programs and modules stored in other directories into the current project.

#### Information on the module library

For special tasks, FST offers an extensive module library with pre-compiled modules. These are mainly programmed in C.



Detailed information on the individual modules of the FST library appears in Volume 2.



#### Note

There is a separate library for each controller type. The corresponding directory is configured automatically. If you wish to change the controller type at a later date, you must reimport the modules from the library for the current controller type.

## 2. Basic functions of the FST software

If you then change back to the original controller type, the previously imported module continues to exist as the imported modules for each controller type are stored separately.

Import a program or a module as follows:

1. Select [Program] [Import...] or the [Import] command in the module or program context menu of the project window. The following dialog then opens, in which you can select the module or the program.

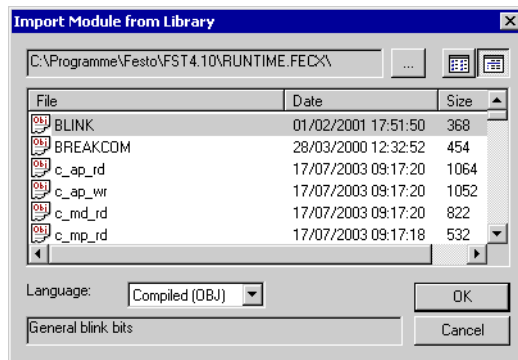


Fig. 2/46: Import program or module

2. If you first want to select the directory in which the program or module is located, click on the “...” button. Select the required directory and confirm with “OK”.
3. From the “Language” list field, select the programming language in which the program or module to be imported was created. If you wish to import a module from the library, select “Compiled (OBJ)”.
4. Now select the file to be imported and confirm your selection with “OK”. The following dialog will then appear.

## 2. Basic functions of the FST software

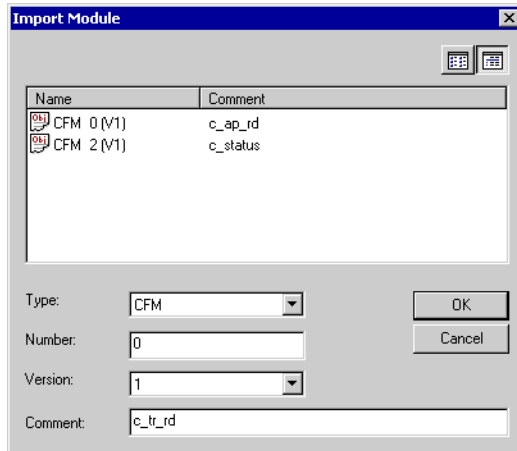


Fig. 2/47: Import program or module

5. As when creating a program or module (see also Section 2.8.1), now select the properties with which the program or module is to be imported into the current project.



When modules are imported the original filename of the module is automatically used as the comment.

6. Confirm the selected properties with “OK”. The program and/or module is then imported into the current project.

### 2.8.3 Exporting programs and modules into the library

You have the option of exporting self-generated modules to the library so that they can be re-used in another project at a later date. You can store modules as a source code or - as with standard modules – in an already compiled OBJ format.

Export programs and modules as follows:

1. Select [Program] [Export...]. A list appears containing all program and function modules in the project.
2. Double-click to select the required module or select it and click the “OK” button.
3. Select whether the module is to be an executable module (OBJ) or is to be stored in the source code. If you select OBJ, the module is stored in the library for the current controller type. Source code modules are stored in a separate LIB directory.

You can change the directory by clicking on the “...” button next to the path display.

4. To save the module, enter a name and click on “OK”.

### 2.8.4 Viewing and changing program and module properties

You can assign an alternative number to an existing program or module or change its version number, type or comment. Define these properties when you create a new program or module. They can be viewed and overwritten at any time.

- To open the properties, select the corresponding element in the program tree window and select the [Properties...] command from the context menu. The properties will then appear.

**1** Only with modules:  
Parameter index

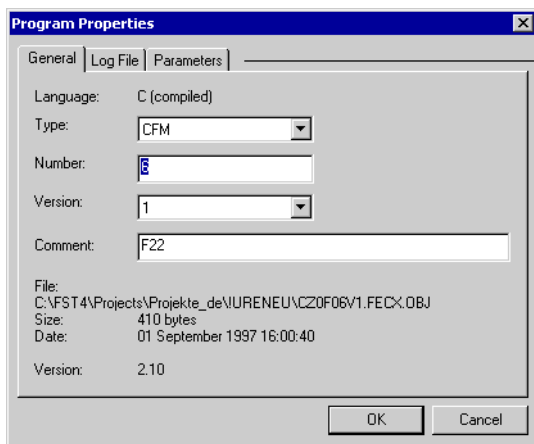


Fig. 2/48: “Program Properties” (in this case module type)



The properties of an open program can be opened using the [View] [Properties] menu command or the context menu in the Editor.

The following information is displayed when certain properties are created (see also Tab. 2/23):

2. Basic functions of the FST software

Index	Description
General	File: File name with directory path Length: File size Date: File date Last compilation: Date on which the program was last successfully compiled (programs only) Version: Version number of the compiled module (modules only)
Log File	For self-generated programs and modules: – Indicates whether the last compilation was successful or not. For modules from the library: – Indicates file and path names of the source file in the library.
Parameters *) (modules only)	Enables a brief informative text on the input and output parameters of the module to be entered or indicates this information.
*) This information is used internally to enter module calls into a program.	

Tab. 2/24: Index in the “Program Properties” dialog window

2.8.5 Deleting programs or modules

You can delete programs or modules that are no longer needed from the current project, e.g. if you have created several test programs that you no longer require.



**Note**  
When a program or module is deleted, the corresponding file is removed from the data carrier.  
Make sure that you no longer need the file. If necessary, create a backup copy beforehand (e.g. by exporting).

## 2. Basic functions of the FST software

Delete a program or module as follows:

1. Select the [Delete] command from the project window in the context menu for the relevant program or module and confirm the prompt. The program or module will then be deleted.

or

1. Select the [Program][Delete...] menu command. All programs and modules for the current project will then appear.

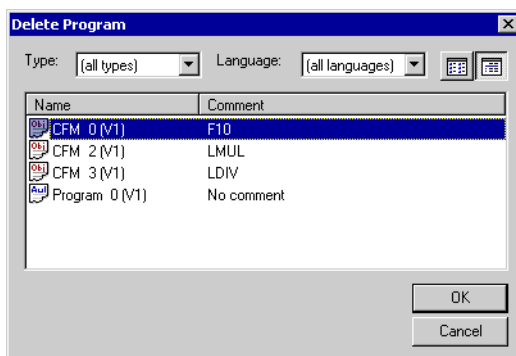


Fig. 2/49: Delete Program

2. You can limit the number of programs and modules displayed using the “Type” and/or “Language” list fields.
3. Double-click on the program or module you wish to delete or select and confirm with “OK”.
4. Confirm the prompt. The program or module will then be deleted.

### 2.8.6 Opening a program or module

To make changes to a source code program or module, you must open its editor window. Open editor windows are displayed within the FST program window.

Open a program or module as follows:



1. Double-click on the relevant symbol in the project window. The corresponding program will then open.

or

1. Select the [Program] [Open...] menu command. The “Open Program” dialog will then appear.

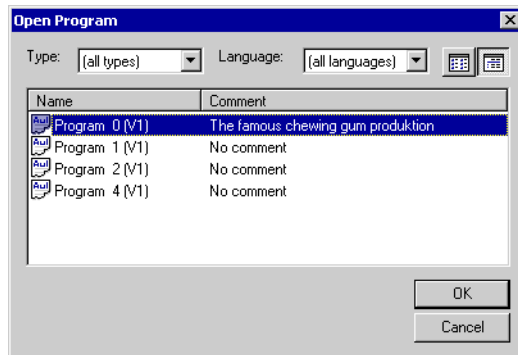


Fig. 2/50: Open Program

2. You can limit the number of programs and modules displayed using the “Type” and/or “Language” list fields.
3. Double-click on the program or module you wish to open or select and confirm with “OK”. The new program and/or module is then opened and displayed in the corresponding editor window.



### 2.8.7 Compiling programs and modules

Compiling involves translating the programs and modules into the internal command syntax of the controller (machine code). This process prepares the data for the controller and they can then be loaded. During compilation, a syntax check is made automatically. Any errors are displayed in a message window and compilation is halted. In this situation, first rectify the errors and then re-start the compilation process.



Only projects that contain no syntax errors can be compiled and loaded.

Compile an individual program or module as follows:

1. Open the program as described in Section 2.8.6.



#### **Note**

Prior to compilation, the relevant program is saved automatically.



2. Select [Program] [Compile]. The program will then be compiled.

If an error occurs during compilation, the operation is cancelled. The error is displayed in bold font in the message window. Double-clicking on an error in the message window takes you directly to the relevant location in the source code. You can then correct the error. Use F4 to skip to the next error.

The result of the compilation is also entered into the error list for the program (for the program properties in the “Log File” tab, see also Section 2.8.4).

## 2. Basic functions of the FST software

### 2.8.8 Selecting an object for compilation and downloading

When projects are compiled and downloaded, only selected programs and modules are involved.

Those programs and modules selected are indicated by a checkbox before the corresponding entry in the project window. A tick in the checkbox means that the corresponding object has been selected.

- 1 Selected programs and modules

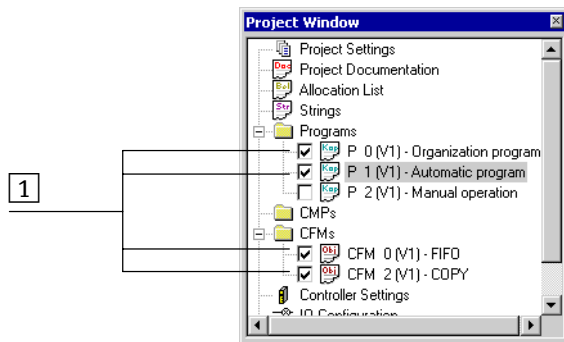


Fig. 2/51: The Project Window shows the selected objects

Select a program or module for compilation and loading as follows:

1. Click the relevant checkbox in the project window. The tick before the program name is applied or deleted.
- or
1. Select the [Program] [[Select for Download] men command and make your selection by clicking the checkbox in the corresponding dialog window.

### 2.9 Online mode

In online mode, a communication link is set-up between computer and controller. You can set-up an online link via the following connections:

- RS232 port
- Ethernet connection (TCP/IP link).



To be able to set-up an online link via TCP/IP, a project with the TCP/IP driver must first have been downloaded to the controller via the RS232 port.

#### RS232 link

Controller and computer are usually linked in series via the RS232 port. This involves creating a simple point-to-point link with a zero modem cable via the COM port on the computer and the COM port on the controller.

Select the COM port on the controller from the PLC settings (Controller Settings) in the “Options” tab. The default setting is the RS232 port on the central unit of the controller (COM or COM1).

Select the COM port on your PC from [Extras] [Preferences...] in the “Communication” index (see Section 2.9.1). The default setting is COM1.

#### TCP/IP link

You can set-up a TCP/IP link with an Ethernet cable via the Ethernet connection. The TCP/IP link has several advantages over the serial connection, e.g. higher transfer speed and the option of linking several controllers and PCs together in one network.

To configure the online link, you must indicate or select the IP address of the controller (see Section 2.9.1). Make sure that PC and controller are connected to the same network.

## 2. Basic functions of the FST software

### 2.9.1 Configuring the online link



#### Note

If you change settings for the online link, all opened online windows will close automatically.



Remember when selecting the baudrate that not all controllers can operate at the same speed. 9600 is generally a good selection. Reduce the baudrate if you do not want to set-up a communication. Further information appears in the documentation for the controller.

Configure the online link as follows:

1. Select [Extras] [Preferences...].
2. Select the “Communication” index.

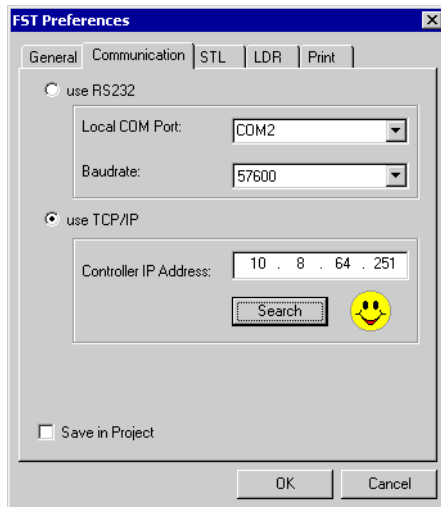


Fig. 2/52: Settings for online communication

## 2. Basic functions of the FST software

### 3. Make the settings required for the communication methods.

- For a link via RS232, indicate the COM port on your PC and the baudrate.
- For a TCP/IP link, indicate the IP address of the controller. Press the “Search” button to search all controllers present in the local network. A dialog then offers for selection a list of all found controllers.

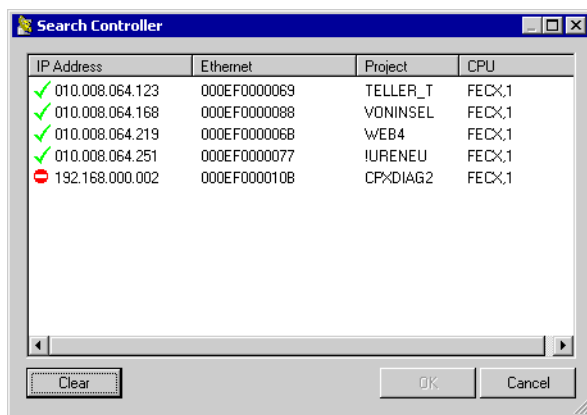


Fig. 2/53: Search controller (example)

- Press “OK” to use the IP addresses of the selected controller as the setting.
  - If the setting is to be valid only for the current project and not for FST in general, tick the “Save in Project” checkbox.
4. Press “OK” to apply the settings.

### Testing an online link

You can test the link to the controller as follows:

- Select [Online] [Login]. The following dialog window provides information on the login process.

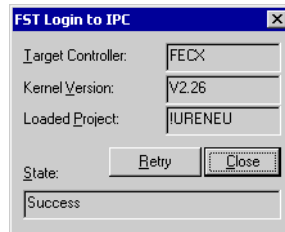


Fig. 2/54: FST Login

If the process is successful, information on the controller is displayed. You can restart the login process by pressing the "Retry" button. Press "Close" to close the dialog again.

This dialog is also displayed when other online functions are activated. In these cases, it closes automatically.



## 2. Basic functions of the FST software

### 2.9.2 The online control panel

The online control panel offers access to important basic functions and information on commissioning and diagnosis (e.g. starting, stopping and halting the controller).

Open the online control panel as follows:



- Select [Online] [Control Panel]. The online control panel will then appear.

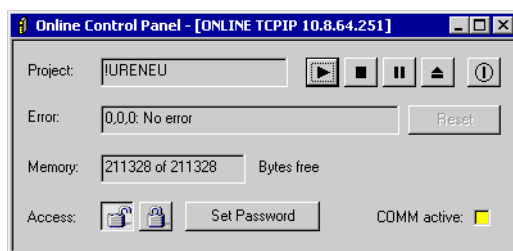


Fig. 2/55: Online control panel








Info fields	Description
Project	Name of the current project in the controller, if “none” is shown, no project has been downloaded.
Error	Error status of the controller with brief description. The following are displayed: <ul style="list-style-type: none"><li>– General errors: &lt;Error number&gt;&lt;Program number&gt;&lt;Step number&gt; <sup>1) 2)</sup></li><li>– CPX error (42): &lt;42&gt;&lt;CPX error number&gt;&lt;Module number&gt; <sup>3)</sup></li><li>– I/O error (11, 12): &lt;Error no.&gt;,&lt;255&gt;,&lt;No. of input or output word&gt;</li></ul>
Memory	Indicates the memory currently available compared with the memory available after the driver and the runtime system were downloaded
Access	Indicates via the buttons with the locks whether the controller is password-protected (see Tab. 2/26).

<sup>1)</sup> If the program has no steps (e.g. LDR programs), step 0 is displayed.  
<sup>2)</sup> Further information on the general FEC errors appears in Section 5.5.5.  
<sup>3)</sup> Information on CPX errors appears in the CPX system manual.

Tab. 2/25: Info fields of the online control panel

## 2. Basic functions of the FST software

The following switch functions are available:

Switch	Description
	Start/continue project; program 0 and all halted programs are activated – in dependence of the PLC settings.
	Stop project; all programs are deactivated and all outputs rest – in dependence of the PLC settings.
	Halt project; program processing is interrupted. All active programs are halted – in dependence of PLC settings.
	Remove project from the working memory of the controller; the project and all drivers are stopped. If you want to reactivate the project, you must reload it or boot the controller.
	Boot the controller (restart)
	Activate password protection for online access
	Deactivate password protection for online access Open a window by having to enter the password to deactivate password protection

Tab. 2/26: Online control panel switches

Buttons	
Default settings	Deletes errors in the controller
Password set/change	Enables the password in the controller to be set or changed The lettering on the button reads: “Set password”, if no password is set in the controller. “Change password”, if a password is set in the controller. If no password is set, the disable can be deactivated without having to make an entry.

Tab. 2/27: Buttons of the online control panel



You have the option of saving a password with the project (see Fig. 2/37).



## 2. Basic functions of the FST software

### Password protection

If the controller is password-protected, you must enter a password to obtain full access to change the operands. The dialog for entering passwords is offered at login.

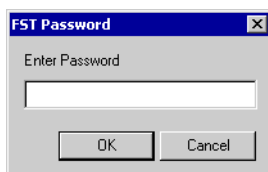


Fig. 2/56: FST password



If no password is set, the access lock on the online control panel can be removed without entering a password.

### 2.9.3 Setting password through the online control panel

You can enter, change or delete a password through the online control panel.



A password consists of between 3 and 20 visible ASCII characters. Separators such as commas, spaces, tab, IBM extended characters etc. are not permitted. Further information on password protection can be found in Section 2.5.4.

Set the password through the online control panel as follows:

1. Select [Online] [Control Panel]. The online control panel will then appear.
2. Click on the “Set password” or “Change password” button. The “Set password” or “Change password” dialog will appear.

Fig. 2/57: Change FST password

3. If you want to change the password, enter the old password into the “Old Password” field.
4. Enter the new password into the “New Password” field. Leave the input field empty, if you want to delete the old password.
5. In the “Again” field, enter the new password and confirm with “OK”.

### 2.9.4 The online display

The online display enables you:

- to display and change current operand values and strings
- force inputs and outputs.

Start the online display as follows:



- Select [Online] [Online Display]. The following dialog will then open.

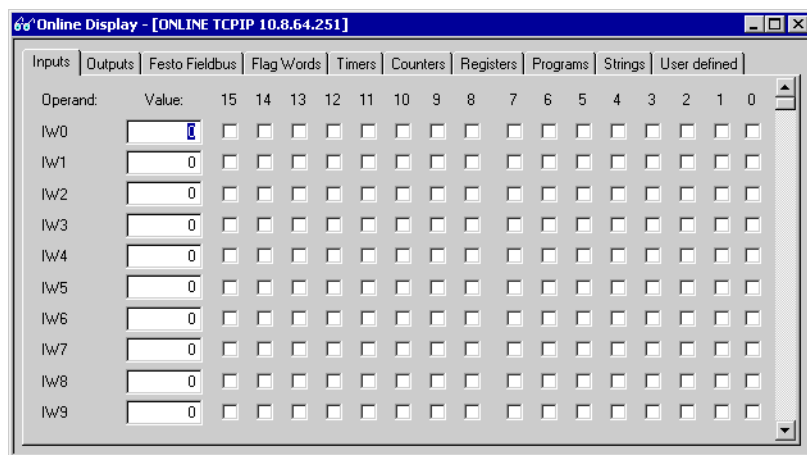


Fig. 2/58: Online display

The various operands are located in the respective indexes. The strings index contains any existing strings.



If you wish to view various operand types and strings in parallel, you can open several online displays at the same time. In the "User defined" index, you can group together any combination of individual operands for the online display.

## 2. Basic functions of the FST software

The bit values of the operands (if addressable bitwise) are displayed in 16 checkboxes. Set single-bit operands (logic 1) are displayed by ticking the checkboxes.

If the mouse pointer is over the operand value or a checkbox, the symbolic identifier of the corresponding operand is displayed. The operand area can be set using the picture scroll bar or the picture keys.

You can change the following for the display:

- the update speed
- for multi-bit values, select the display format (decimal with prefix, decimal without prefix or hexadecimal).

### Changing the update speed

You can set the waiting time between two requests for the online display. The set “Update Speed” is effective in all online windows of the same type, e.g. in all online display windows.

Change the update speed as follows:

1. Select [Online] [Change Update Speed...].

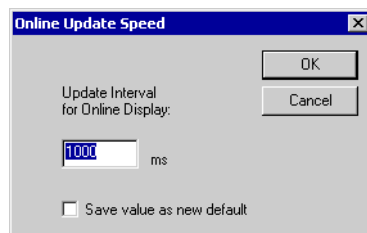


Fig. 2/59: Online update speed

2. Enter the required delay time in milliseconds. Times between 10 ms and 10000 ms can be entered, the default value is 1000 ms.

## 2. Basic functions of the FST software

### Setting online display format

Multi-bit values can be displayed in the following formats:

- decimal with sign
- decimal without sign
- hexadecimal.

Set the required format as follows:

- Select the desired format from the menu [Online] or from the context menu of the online display. All multi-bit values are then displayed in the selected format.

### User defined list

In the “User defined” index, you can group together a list of operands for the display. The error word (EW) or the error status (E) are also permitted.

- To configure a new operand, click on an empty button in the “Operand” column. A dialog opens in which the operand is entered or can be selected from the allocation list.

The operands configured in the “User defined” index are stored in the project as absolute operands.

### Festo fieldbus

The “Festo Fieldbus” index indicates which users are present when the index is first activated on the fieldbus. If the configuration in the master has been changed, activate “Refresh Display” from the context menu to update the user list.

The “Festo Fieldbus” index indicates the users present on the fieldbus and enables the configuration to be requested and assigned in the master.

## 2. Basic functions of the FST software

- To update the configuration in the fieldbus master to reflect the users present on the fieldbus, select “Record actual configuration” from the context menu. Please note that this will temporarily uncouple all users from the master and all fieldbus outputs will be reset.
- To assign the nominal configuration (i.e. the user list known for the programs), select “Assign nominal configuration” from the context menu. Please note that this will temporarily uncouple all users from the master and all fieldbus outputs will be reset.

### Strings

The “Strings” tab enables you to view and change the strings.



If you wish to use strings in a FST IPC project, you must enter and parameterise the STRINGS driver in the driver configurator (see Volume 2).



#### **Note**

For technical reasons, it may not be possible to display or change, either partly or completely, strings with special characters in the online display. This applies specifically to the following special characters: \0, \a, \n, \r, \11, \14.

- Use these special characters in the online display to prevent the strings being manipulated.



Special characters in strings may have an undesirable influence on communication via RS232 and/or TCP/IP, if these characters are significant in the transfer protocol. TCP/IP communication is less sensitive than communication via RS232.

### Goto operand

In the online display, you can skip to an operand.

1. Select [Online] from the menu or the [Goto...] command from the context menu of the online display.

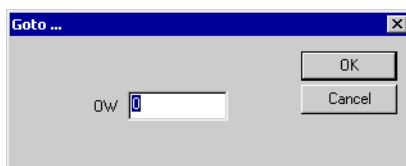


Fig. 2/60: Goto...

2. Enter the number of the required operand and confirm with "OK". The displayed operand area is then changed so that the required operand appears at the top.

### Changing operand value



#### **Warning**

Depending on the function of the machine/system, manipulation of signal states can cause serious physical injury and damage to property.

- Exercise extreme caution when using the change function in order to avoid damage.

The following methods are available for changing an operand value:

- Double click on a multi-bit value (word), a timing or a string, in order to open the Change operand dialog (and/or change timer or string).
- Click on the checkbox of a bit to change its value.
- Click on the checkbox of a timer to start or stop it.

## 2. Basic functions of the FST software

- With programs, select the desired status (active, inactive, halted) from the selection box. Please note that programs can be halted only from the active status.

### Changing single-bit and multi-bit operands

Change the value of a single-bit or multi-bit operand as follows:

1. Move the mouse pointer over the current value and select the [Modify Operand] command from the context menu. Depending on the operand type, a corresponding dialog will appear for modification, e.g.:

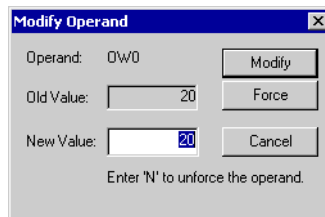


Fig. 2/61: Change operand

2. Enter the required value and confirm with “Modify”. The value is then transferred to the controller.

### Changing single-bit operand

The value of a single-bit or operand can also be changed as follows:

- Click on the relevant checkbox. The value will then be changed accordingly.



### Forcing inputs and outputs

#### Forcing

Forcing can be used to enforce input and output signals. Input signals actually present or changes of status by program will be ignored and replaced by the force values.

#### Forcing inputs

Forcing an input does **not** change the input signal and can also **not** be observed on the corresponding status LED. The logical status of the input is modified internally and in some cases has an effect on the program. The forced input status is transferred to the input process image. The online display therefore shows the forced input signal.

#### Forcing outputs

By contrast, forcing an output changes the actual output signal and can be observed on the corresponding status LED. The forced output status is, however, **not** transferred to the output process image. The online display does **not** show the forced, physical output signal. Instead it shows the status from the process image.



#### Note

The online display always shows the signal status valid in the process image. You should therefore observe the following when forcing:

- Forced input statuses are transferred to the process image and therefore detected by the controller. These are visible in the online display.
- Forced output statuses are **not** transferred to the process image and therefore **not** detected by the controller. They are therefore **not** shown in the online display.

### Information on forcing with the CPX terminal

With the CPX terminal, forcing is globally enabled or disabled for the CPX terminal via the force mode system parameter. The current parameter settings are displayed in the context menu of the online display. A tick in front of the command [Enable Force Table] indicates that forcing has been globally enabled.

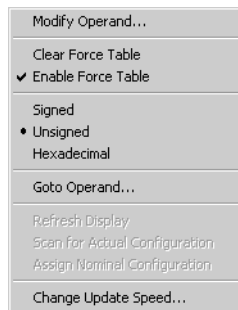


Fig. 2/62: Enable force table (for CPX terminal only)



#### Note

If you globally disable or enable forcing, all signal statuses enforced by forcing may be immediately rendered invalid or valid respectively.

If the force mode system parameter is changed, the channel-oriented force settings (force mode and force state module parameters) will be reset automatically in the following cases, in order to prevent undesired signal states:

- Changing via FST: Change from enable to disable.
- Change via fieldbus: Change from disable to enable.

## 2. Basic functions of the FST software



If the current force settings are not known, you should first create a clear starting status for the forcing, e.g. by deleting all channel-oriented force settings.

Globally deleting force values

The online display enables the channel-oriented force settings to be globally deleted for the entire controller.



### Note

If you globally delete the channel-oriented force settings (force table), all enforced signal statuses immediately become invalid. All signal statuses from the process image are again valid.



### Note

With the CPX terminal, the global deletion of force values returns the following channel-specific module parameters to their factory settings:

- Force mode: 0 (forcing disabled for this channel)
- Force state: 0 (force value for this channel 0)

Delete the channel-oriented force settings via the online display as follows:

- Select the [Clear Force Table] command from the context menu of the online display (see also Fig. 2/62).

Globally enable/disable forcing (CPX-FEC only)

The online display enables forcing to be globally enabled or disabled for the CPX terminal.

You can use the online display to globally enable or disable forcing:

- Select the [Enable Force Table] command from the context menu of the online display.

## 2. Basic functions of the FST software



With the CPX terminal, forcing can also be enabled using the hardware configurator (see Section 2.3.2). Further information on forcing with the CPX terminal appears in the CPX system manual.

### Changing force values using the online display

#### Changing force values

Change the force values of inputs and outputs as follows:

1. Move the mouse pointer over the current value and select the [Modify Operand] command from the context menu. Depending on the operand type, a corresponding dialog will appear, e.g.:

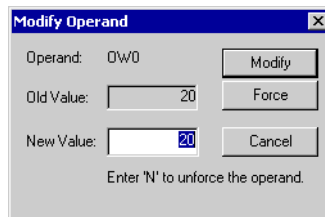


Fig. 2/63: “Modify Operand” dialog window



#### Note

With the CPX terminal, the deletion of a force value returns the following channel-specific module parameters to their factory settings:

- Force mode: 0 (forcing disabled for this channel)
- Force state: 0 (force value for this channel 0).

2. Enter the desired value. Enter “N” as a new value if you wish to deactivate forcing for this value again.
3. To transfer the value to the controller, confirm with “Force”.

## 2. Basic functions of the FST software

### 2.9.5 FST file transfer

For handling files in combination with the controller, FST offers the “FST File Transfer” window. This provides the following functions:

- Upload files from the controller
- Delete files from the controller
- Download files to the controller
- Display directory and files from the controller drives
- Display file size and creation date
- Display the free memory area on the controller drives

Open the FST file transfer window as follows:



- Select the [Online] [File Transfer] menu command. The “FST File Transfer” window will then appear.

- 1 Path name  
(current folder)
- 2 Function bar
- 3 Transfer status  
(red = transfer  
running)
- 4 Transfer status  
(green = transfer  
complete)
- 5 Status line

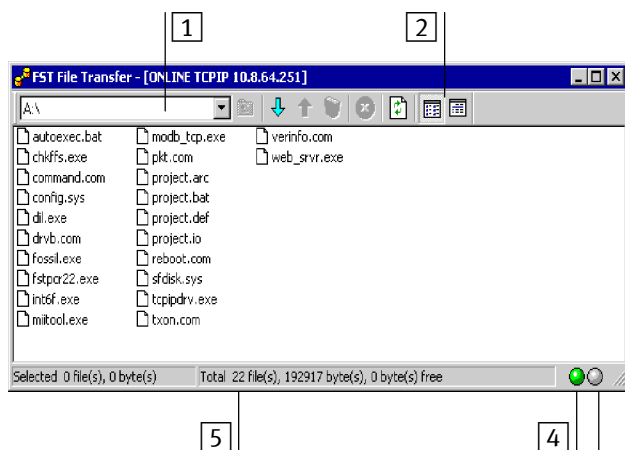










Fig. 2/64: FST file transfer window

## 2. Basic functions of the FST software

All files and sub-directories of the current working directory of the controller are displayed. The path name (in this case A:\) is displayed in a single-line list field on the function bar. The sub-directories (folders) it contains are identified by a folder symbol and the files with a page symbol.

The status line in the window indicates the size of the selected file, the overall size of all files and the free memory area on the drive.

The following commands are available on the function bar:

Icon	Meaning	Description
	Higher-order directory	Go to higher-order directory
	Download	Download files from the computer to the controller
	Upload	Upload selected files or display content of a sub-directory
	Delete	Delete selected files from the controller
	Stop	Cancel data transfer
	Update	Update view (current directory is uploaded and displayed again)
	List	View as list; file names only displayed as list
	Details	View with details; file name, file size and created date displayed as list

Tab. 2/28: Switch fields on function bar of FST file transfer

### Change view

- The “Details” and “List” buttons can be used to toggle between the views as required. The “Update” button enables you to upload and display the current directory again.
- If you want to display the content of a sub-directory, double-click the directory name or select it and click on the “Upload” button.
- Clicking on the “Higher-order directory” button displays its contents.
- If you want to change the drive, select the new one from the selection box on the function bar.

### Uploading a file from the controller

During the data transfer operation, the red lamp in the status line illuminates and the current transfer status is displayed. You can use the “Stop” button to cancel the current data transfer operation.

#### Uploading a file



Upload a file from the controller as follows:

1. Double-click on the required file or select it and click on the “Upload” button.
2. In the Windows standard dialog, specify the target directory and where necessary a new file name under which the file is to be saved on your computer.

## 2. Basic functions of the FST software

### Downloading a file to the controller

#### Download file

Download a file to the controller as follows:



1. In the “FST File Transfer” window, open the directory to which the file is to be transferred.
2. Click on the “Download” button.
3. In the Windows standard dialog, select the required file and confirm with “OK”. The file is then transferred to the current directory in the controller.



During the data transfer operation, the red lamp in the status line illuminates and the current transfer status is displayed. To cancel the current data transfer, click on “Stop”.

### Deleting files from the controller

#### Deleting a file

Delete a file from the controller as follows:



1. Select the required file and press the DEL key or click on the “Delete” button.
2. Confirm the prompt with “Yes” to permanently delete the file.

To restore communication, you must connect the controller to the computer and possibly configure the communication interface.

File transfer will not function correctly if the controller is password-protected (disabled).



## 2. Basic functions of the FST software

### 2.9.6 The command interpreter terminal

FST handles communication with the controller operating system via the command interpreter (CI). The CI of the operating system knows numerous commands for displaying and modifying operands, starting and stopping programs, transferring files etc.

The FTS online displays also operate with these CI commands. If you wish to manually send CI commands to the command interpreter, use the CI terminal window.

Opening the CI terminal

Open the CI terminal as follows:



- Select [Online] [Terminal]. The CI terminal window will then open.

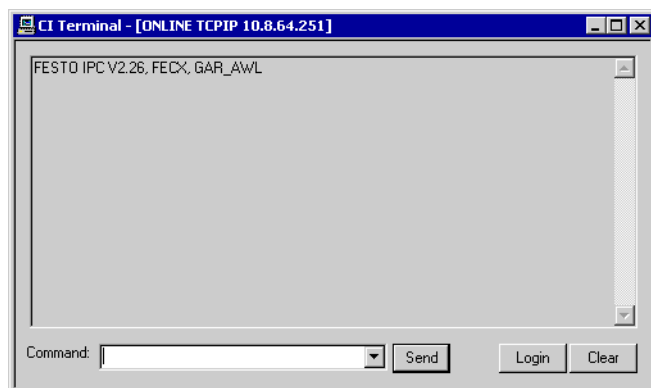


Fig. 2/65: CI terminal

## 2. Basic functions of the FST software

### Sending a CI command

Send a CI command to the controller as follows:

1. Enter the CI command into the “Command” input field.
2. Close the enter by pressing Enter or click on the “Send” button. The command is then sent to the controller. Your command and the response from the controller are displayed in the display field above the edit field.
3. The CI command remains in the input field.
  - If you wish to send this command again, repeat Step 2.
  - If you wish to delete the content of the display field, click on the “Clear” button.
  - To start a new login, click on the “Login” button. If successful, the version number of the main program runtime, the controller type and the name of the active project are displayed.

### 2.9.7 Online mode of the hardware configurator (FEC-CPX only)



Information can be found in section 2.3.2.

### 2.9.8 STL and LDR Online display



Information can be found in Sections 3.5 (STL Online Display) and 4.4 (LDR Online Display).

## 2. Basic functions of the FST software

### 2.9.9 The Web Browser

You can call up web pages in your controller using the web browser integrated into FST. Requirement:

- Ethernet link between controller and PC
- Web server driver has been loaded.



Further information on setting-up an Ethernet link and the web server driver appears in Volume 2.

Opening the web browser    Open the integrated web browser as follows:



- Select [Online] [Web Browser]. The web browser will then open.

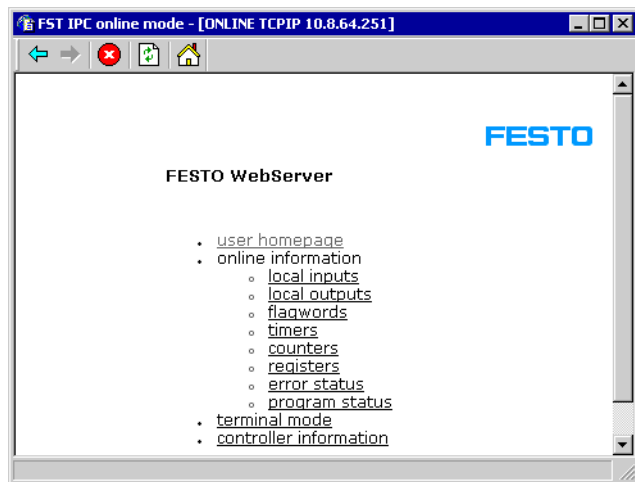







Fig. 2/66: Web browser with standard homepage index.htm

## 2. Basic functions of the FST software

The following commands are available on the function bar:

Icon	Meaning	Description
	Back	Scroll back to previous web page
	Forward	Scroll forward to next web page
	Stop	Cancel data transfer
	Update	Update view (current web page is uploaded and displayed again)
	Start page <sup>1)</sup>	Display Main.htm stage page, if present. Display Index.htm standard homepage, if Main.htm is not present.
<sup>1)</sup> The web server driver already contains standard HTML pages. The standard homepage is called Index.htm. Call the homepage you generate yourself Main.htm.		

Tab. 2/29: Switch fields on function bar of web browser

### 2.10 Runtime library (drivers and I/O scripts)

Supplied with FST are a host of descriptions on input and output modules (I/O scripts) and drivers. These are summarised in a runtime library and automatically installed by the installation program. The drivers and I/O scripts are stored separately for each controller type.

#### I/O scripts

The I/O scripts enable the hardware configuration of the controller in use to be replicated. For the CPX terminal, FST contains a special hardware configurator (see Section 2.3.2). No I/O scripts for this type are therefore managed in the runtime library.

#### Drivers

Drivers expand the functionality of the FST-PLC operating system for the relevant controllers. Standard drivers are available for each controller type.



Detailed information on the drivers and modules available for the various controllers appears in Volume 2.

#### Managing runtime library

You can use FST to conveniently manage the runtime library. You can:

- retrospectively install (to support new hardware and new functions)
- extract from the runtime library (e.g. for subsequent installation)
- and remove drivers and scripts from the runtime library (de-install).

2. Basic functions of the FST software

You must open the runtime library before you can make changes.



**Note**  
Changes made in the FST library affect all projects. If you deinstall I/O scripts or drivers that are already being used in projects, you can no longer download these projects to the controller without re-installing the I/O scripts and drivers.

Open runtime library

Open the runtime library as follows:

- Select [Extras] [Library...]. The “FST Library” dialog is then opened.

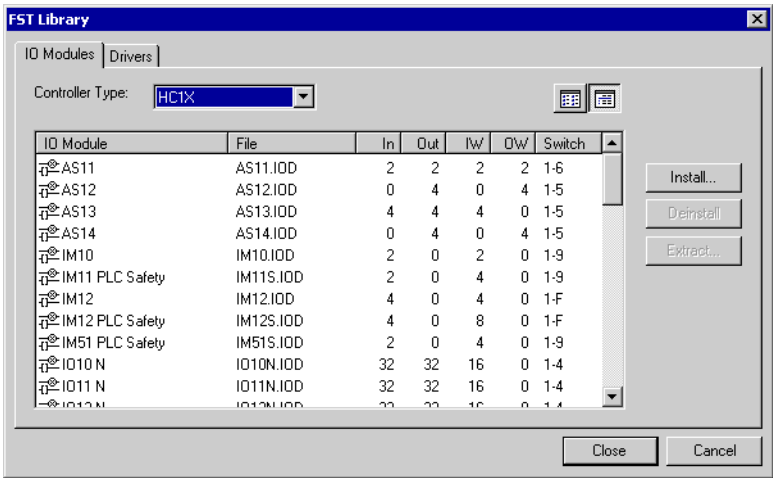


Fig. 2/67: FST runtime library (I/O cards index)

The display format can be changed, as is the norm with multi-column list fields, by clicking on the corresponding button in the top right-hand corner.

## 2. Basic functions of the FST software

### 2.10.1 Managing I/O scripts in the runtime library

The I/O scripts are managed in the “FST Library” dialog in the “IO Modules” index (see Fig. 2/67). The multi-column list field indicates the installed I/O scripts (I/O cards) for the selected controller type. Meaning:

Column	Description
IO Modules	Script description
File	File name of the I/O script (I/O scripts have the file name extension *.IOD)
In	Number of occupied input bytes in the I/O area of the processor
Out	Number of occupied output bytes in the I/O area of the processor
IW	Number of input words occupied by the card
OW	Number of output words occupied by the card
Switch	Possible switch position for setting the I/O area

Tab. 2/30: Columns in the “IO Modules” index

#### Installing a new I/O script

- Installing a new I/O script
- Open the runtime library and then install a new I/O script as follows:
1. From the list field, select the controller type for which you would like to install an I/O script.
  2. Press the INS key or click on the “Install...” button. A dialog for selecting the script file will then open.

## 2. Basic functions of the FST software

3. From the Windows standard dialog that then appears, select the directory and file names of the script file and confirm the selection. The script file is then installed and available for projects.

### Extracting an I/O script (copying)

You can copy I/O scripts to any data storage device, e.g. for subsequent installation.

#### Extracting an I/O script (copying)

Open the runtime library and then extract an I/O script as follows:

1. Select the controller type from the list field.
2. Select the I/O script you want to copy.
3. Click on the “Extract...” button.  
A dialog for selecting the data storage device and the target directory will then open.
4. Select the data storage device and the target directory and confirm the selection. The I/O script is then transferred.

### Deinstalling an I/O script (deleting)

#### Deinstalling I/O scripts

Delete a selected I/O script from the library as follows:

- Click on the “Deinstall” button or press the DEL key. The script file will then be deleted.



## 2. Basic functions of the FST software

### 2.10.2 Managing drivers in the runtime library

The drivers are managed in the “FST Library” dialog in the “Drivers” index (see diagram below).

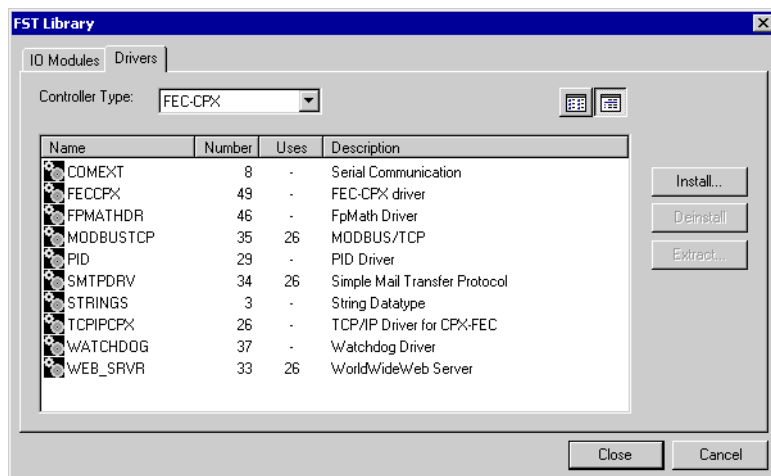


Fig. 2/68: FST runtime library (drivers register)

The multi-column list field indicates the installed drivers for the selected controller type. Meaning:

Column	Description
Name	Driver name (driver scripts have the file name extension *.dmk)
Number	Driver number
Uses	Number of other drivers required
Description	Brief description on the driver

Tab. 2/31: Columns in the drivers index

## 2. Basic functions of the FST software

### Installing new drivers

#### Installing drivers

Open the runtime library and then install a new driver as follows:

1. From the list field, select the controller type for which you would like to install a driver.
2. Press the INS key or click on the “Install...” button. A dialog for selecting the driver file will then open.
3. In the standard Windows dialog that appears, select the directory and file name for the driver script file and confirm your selection. The driver is then installed and available for projects.

### Extracting drivers (copying)

You can copy drivers to any data storage device, e.g. to transfer it to another FST installation.

#### Extracting drivers (copying)

Open the runtime library and then extract a driver as follows:

1. Select the controller type from the list field.
2. Select the driver you wish to copy.
3. Click on the “Extract...” button. A dialog for selecting the data storage device and the target directory will then open.
4. Select the data storage device and the target directory and confirm the selection. The driver and the corresponding driver script file are then transferred.

## 2. Basic functions of the FST software

### De-installing drivers (delete)

#### De-installing drivers

Delete the selected driver and all corresponding modules from the library as follows:

- Click on the “Deinstall” button or press the DEL key. The driver, the driver script file and all corresponding modules are then deleted.

### 2.11 External Tools

External applications can be started directly from the FST interface. At program start, FST-specific parameters can be transferred, e.g. name of the current project. The program calls, that you manage with FST, are entered into the [Extras] menu. The tools can be started from this menu. You can:

- insert, delete and edit new program calls
- move program calls into the [Extras] menu.



Only the first 25 calls are entered into the “Extras” menu.

Opening management dialog

Open the dialog for managing the program calls as follows:

- Select [Extras] [Configure Tools...]. The following dialog will then open:

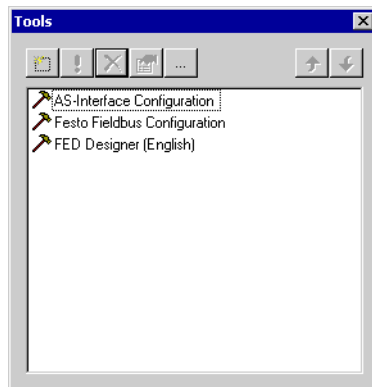








Fig. 2/69: “Tools” dialog

The program calls appear in the dialog in the same sequence as in the [Extras] menu.

## 2. Basic functions of the FST software

Buttons are available above the list:

Icon	Operation
	Insert new program call
	Run tool
	Delete program call
	Change call properties (description, program name, parameters and options)
	Move selected item upwards
	Move selected item downwards

Tab. 2/32: Operations in the “External Tools” dialog

### 2.11.1 Inserting or changing a program call



- To insert a program call for an external tool, click on the “New” button.



- To configure the properties of the selected call, click on “Properties”.

The following dialog will open:

2. Basic functions of the FST software

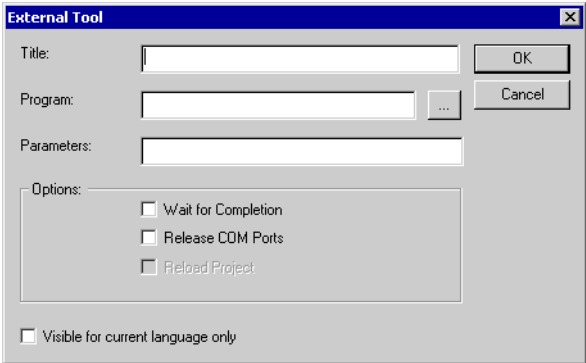


Fig. 2/70: Call properties

Field	Description
Title	Designation to appear in the “Extras” menu. You can tag the subsequent letters for direct access with the & sign. This letter appears in the menu underlined. To insert the & sign, enter “&&”.
Program	Path to program to be run. The “...” button enables you to search your drives. The macros described below can also be used.
Parameters	Parameters to be transferred at program start. The macros described below can also be used.
Wait for Completion	Minimise FST main window. If the option is not selected, the FST main window remains open.
Release COM Ports	Select this option if the external tool uses the same serial port as FST. When the external port has been called, it then occupies the serial port. While this port is active, FST cannot use it to set-up a link.
Reload project	Current project is closed prior to the external tool being called and reopened when the tool is terminated. Changes to the external tool in the project are then directly visible. If you select this option, after the external tool is terminated, the current project is re-opened so that changes made via the tool are visible. The option is valid only if “Minimise FST main window” has also been selected.

Tab. 2/33: Fields in the “External Tool” dialog

## 2. Basic functions of the FST software

The following macros are replaced during program and parameter entry:

Macro	Description
<<<FSTDIR>>>	FST master directory
<<<RUNTIMEDIR>>>	One of the “runtime...” directories in the FST master directory; “<<<FSTDIR>>>\Runtime.<Controller type>,” e.g. “C:\Programmes\Festo\FST4.10\Runtime.FEC”
<<<ALLPROJECTS>>>	Higher-order project directory; presides over the directories containing the projects, e.g. C:\FST4\PROJECTS
<<<PROJECT_CUR>>>	Directory name of the current project, e.g. PROJECT1
<<<CURRENTPROJECT>>>	Rung to the current project (instead of<<<ALLPROJECTS>>>\<<<PROJECT_CUR>>>)
<<<TARGET>>>	Code for current controller type (“FEC”, “FEC2”, “FECX”, “HC0X”, “HC1X” or “HC2X”)
<<<LANGUAGE>>>	Code for current language (“GB” or “D”)

Tab. 2/34: Macros for program and parameter entry

The following macros are replaced only in parameter entry:

Macro	Description
<<<COM_PORT>>>	Port on the computer for the controller (e.g. “COM1”)
<<<COM_BAUDRATE>>>	Baudrate at the COM_PORT (e.g. “9600”)
<<<#1>>> (obsolete)	“VFNM”
<<<#2>>> (obsolete)	“X”
<<<#3>>> (obsolete)	<<<CURRENTPROJECT>>>
<<<#4>>> (obsolete)	“IPC”
<<<#5>>> (obsolete)	<<<COM_PORT>>>“/”<<<COM_BAUDRATE>>>“/9000”

Tab. 2/35: Macros for parameter entry

## 2. Basic functions of the FST software



# **Programming in statement list**

## **Chapter 3**

Contents

**3. Programming in statement list ..... 3-1**

3.1 Fundamental principles for programming in the statement list ..... 3-4

3.2 The statement list editor ..... 3-5

3.2.1 STL editor preferences ..... 3-6

3.2.2 Functions of the STL editor ..... 3-7

3.3 Structure of the STL programs ..... 3-20

3.3.1 Step program ..... 3-21

3.3.2 Parallel logic program ..... 3-23

3.3.3 Executive part ..... 3-24

3.4 Brief description of the STL instructions ..... 3-25

3.5 STL online display ..... 3-39

3.5.1 Functions of the STL online display ..... 3-40

3.5.2 Deactivating STL online display ..... 3-46

### 3. Programming in statement list

Contents of this chapter	<p>This chapter describes the functions of the STL editor and the options of the STL online display. The STL editor is used to create your user programs. The STL online display enables you to display status information and also display and change operand values while the controller program is running.</p> <p>It also contains a brief description of the STL instructions, which provide you with an initial overview of the programs in the statement list.</p>
Further information	<p>A detailed description of STL and LDR operations appear in Appendix A.</p>

#### 3.1 Fundamental principles for programming in the statement list

Statement list, or STL for short, is a text-based programming language. STL enables the programmer to describe the working steps of controller functions to be described by simple instructions. The language set-up supports the efficient handling of complex tasks. A STL program can be described as:

- Step program  
(stepwise program execution, see Section 3.3.1)
- Parallel logic program  
(execution similar to an LDR program, see Section 3.3.2)
- Executive part (for instructions without input conditions, see Section 3.3.3).

Each sentence of a STL program may comprise a condition part and must comprise a run part.

The condition part contains one or several conditions, which may have linked logically or arithmetically “true” or “false” respectively as a result. The condition part is always introduced with IF, which is followed by one or several instructions that are to be called.

If the programmed conditions are true, all instructions in the run part of the sentence are executed. The run part is introduced with THEN.

## 3.2 The statement list editor

The statement list editor is a text editor featuring the usual functionalities of Windows text editors. It also offers the following functions:

- fast call-up of all STL keywords via shortcuts or menu commands
- automatic line formatting
- automatic allocation list entry and insertion of operands and allocation list comments through selection from the list
- dialog-controlled insertion of module calls
- editing functions such as copying, find/replace, undo etc.
- various configuration options for the program display, e.g. font style, font size etc.

In order to create a STL program, you must first build a project (see Section 2.2.1). Any new STL program you insert or existing one you open will be shown in the STL editor window. With new STL programs, the workspace is empty.

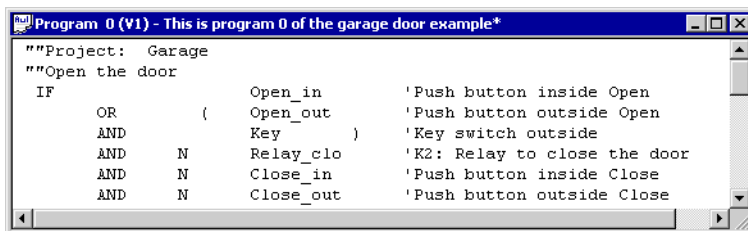


Fig. 3/1: STL editor (example)

### 3. Programming in statement list

#### 3.2.1 STL editor preferences

The [Extras] [Preferences...] command enables you to set the following preferences for the STL editor:

“General” tab (preferences valid for STL and LDR editors):

- automatic activation/de-activation of an allocation list item
- display symbol/operand with allocation list comment

STL tab:

- Font preferences for the STL commands, operands, comments, jump labels etc.

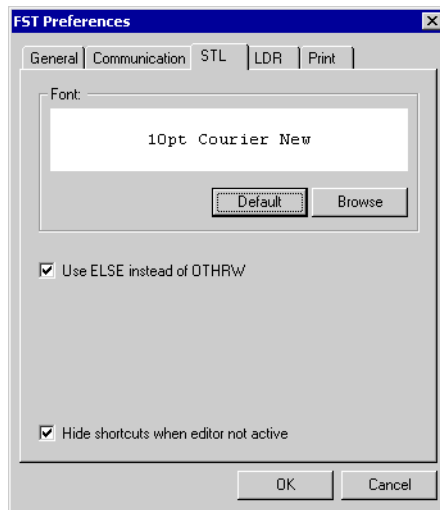


Fig. 3/2: STL editor preferences

The white display field of the “STL” tab displays the name and size of the selected font.

- Click on “Default” to use the Windows default font.
- Click on “Browse”, if you wish to select the font style in the standard Windows dialog.

### 3. Programming in statement list

Use ELSE instead of  
OTHRW

Check this option if you want to use the STL keyword ELSE instead of OTHRW. The default is OTHRW.

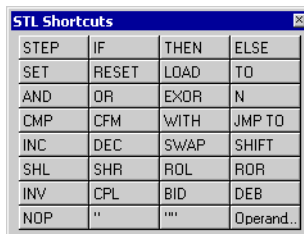
## 3.2.2 Functions of the STL editor

### Inserting STL instructions

You can enter STL instructions directly into the STL editor. You can select the STL keywords from the [Insert] [STL Instruction] menu.

STL shortcut

The STL shortcut facility offers direct access to STL keywords. The shortcut features one button for each keyword. Click to insert the required keyword at a current position.



STEP	IF	THEN	ELSE
SET	RESET	LOAD	TO
AND	OR	EXOR	N
CMP	CFM	WITH	JMP TO
INC	DEC	SWAP	SHIFT
SHL	SHR	ROL	ROR
INV	CPL	BID	DEB
NOP	''	'''	Operand...

Fig. 3/3: STL shortcuts

- To open the STL shortcut, select [View] [Shortcuts] or [Shortcuts] from the context menu.

The shortcut is automatically hidden if the STL editor window is no longer the active window. The shortcut can be placed in any position as a separate window. It can also be moved with the mouse to below the menu bar or arranged on any window edge.

### 3. Programming in statement list

<b>Keyword</b>	<b>Brief description</b>
STEP	Divides programs up into steps (step programs only)
IF	Introduces an operating part
THEN	Introduces the run part of a sentence
OTHRW/ELSE	Introduces an alternative run part that is run when the condition under IF is not satisfied.
SET, RESET	Sets/resets single-bit operand
LOAD	Loads operand or constant into the single-bit or multi-bit accumulator
TO	Transfers in combination with LOAD, the content of the single-bit and/or multi-bit accumulator to Operand 2
AND, OR, EXOR, N	Logical linking of single-bit or multi-bit operands and constants
CMP, CFM	Calls up program (CMP) or function module (CFM)
WITH	Used with CFM/CMP instructions for parameter transfer
JMP TO	Jump to a step label
INC, DEC	Count forwards/backwards (increase or decrease value by 1)
SWAP	Swaps high-byte for low-byte in the multi-bit accumulator
SHIFT	Exchanges single-bit operands for a value in the single-bit accumulator
SHL, SHR	Shift all bits in the multi-bit accumulator one position to the left (SHL) and/or to the right (SHR)
ROL, ROR	Rotate all bits in the multi-bit accumulator one position to the left (ROL) and/or to the right (ROR)
INV, CPL	Single complement / double complement
BID, DEB	Converts binary to decimal display/decimal to binary display
NOP	No operation
“, ““	Short comment/Long comment
Operand	Selecting operand from the allocation list

Tab. 3/1: STL keywords



### 3. Programming in statement list

Observe the following when inserting STL instructions:

- You can use the INS key to toggle between insert and overstrike mode. If insert mode is set, “ON” appears in the status line.
- Every time you leave a line, it is automatically formatted. If syntax errors are found in the line, no formatting is possible and the line remains unchanged.
- If during line formatting an operand is discovered that has not yet been entered into the allocation list, you will be offered the dialog to enter it into the allocation list. If you do not wish to enter the operand into the allocation list, click on “Cancel” or press the ESC key.
- The visibility, size and position of the shortcut windows is saved between the FST sessions.

Insert a STL keyword into the program as follows:

1. Position the cursor where you wish to enter the STL keyword.
2. Click on the required keyword in the STL shortcut. It will then be inserted.

### 3. Programming in statement list

#### Comments

You can make comments in your STL program in one of two ways:

- with short comments
- with long comments.

##### Short comments

Short comments with a maximum of 36 characters are appended to a program line. They begin with a single quotation mark.

##### Example

```
IF      N      I1.7      "no signal from sensor
```

##### Long comments

You can enter long comments via an entire line. They begin with a double quotation mark.

##### Example

```
" "This is a long comment that runs over an  
" "entire line.
```



#### Automatic allocation list entry

The [Extras] [Preferences...] command enables you to activate or deactivate automatic allocation list entry for the STL and LDR editor together in the “General” tab.

If the function is active and you enter a symbolic or absolute operand into the program, which is already entered in the allocation list with a comment, the comment is transferred from the allocation list into the program. Unless, however, you have entered a comment manually. This automatic comment starts with a single quotation mark and cannot be changed in the STL editor.

When an operand is entered, which has not yet been input into the allocation list, you will be offered the dialog to enter it into the allocation list.

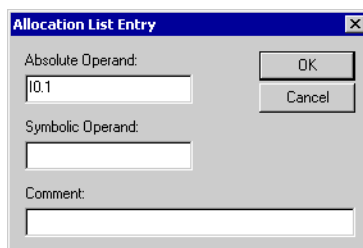


Fig. 3/4: Dialog for automatic allocation list entry

Press “OK” to transfer the operand into the program and the allocation list.

- If you do not wish to transfer the operand into the allocation list, click on “Cancel” or press the ESC key. The operand will then be transferred only to the program.

### 3. Programming in statement list

#### Selecting operand from the allocation list

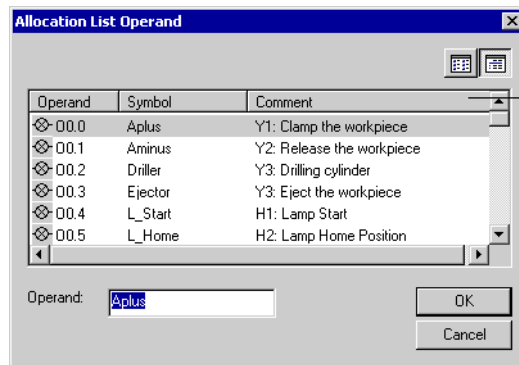
Instead of entering an operand directly into the program, you can select it from the allocation list.

#### Selecting an operand

Select an operand from the allocation list as follows:

Operand...

1. Select [Insert] [Operand...] or click on “Operand” in the STL shortcut. The following dialog will then appear.



#### 1 Table header

Fig. 3/5: Selecting from the allocation list

2. Select the desired operand and confirm your selection with OK. The operand will then be inserted.



You can change the width of the columns in the table header using the mouse. Each column of the window is sorted alphabetically by clicking on the column header. Clicking again reverses the sorting sequence.

#### Inserting module call via dialog

Many modules use the local program function units FU32 to FU38 to exchange parameters with the selecting program. These function units are used both as input and return parameters.



Detailed information on modules and parameters appears in Volume 2.

Use the option of inserting module calls via dialog if you would like to view a brief description on the individual module parameters during entry.

Insert a module call at the current position via the dialog as follows.

1. Select [Insert] [Module Call...] A list appears containing all modules in the project.

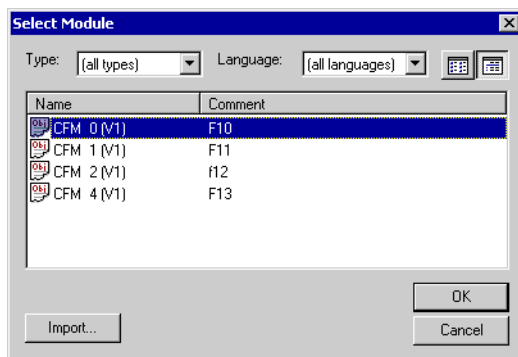


Fig. 3/6: Select module



If you have not yet imported the required module into the project, click on “Import...” (for further information on importing, see Section 2.8.2).

### 3. Programming in statement list

2. Select the required module and confirm the selection.  
The “Module Call...” dialog will then open. An explanatory comment will appear for each module parameter, if a description for the selected module already exists.

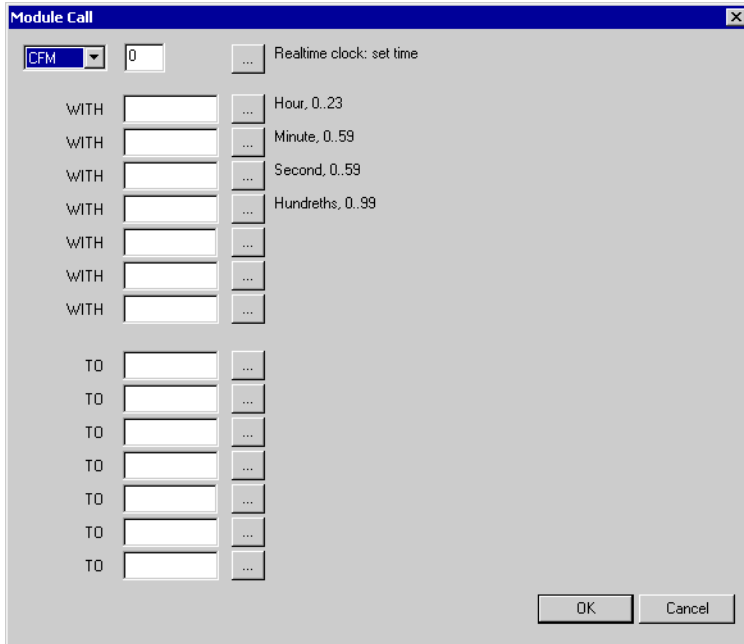


Fig. 3/7: Dialog-based module call

3. If you wish to change the selected module, select an alternative type from the top list field or enter an alternative module number into the input field.

In order to select an alternative module from the list of project modules, click on the top button with the three dots. The “Select Module” dialog will then appear (see Fig. 3/6).

4. If you wish to enter parameters, enter the input parameters under “WITH” and the return parameters under “TO”.

### 3. Programming in statement list

To select an operand from the allocation list as a parameter, click on the button with the three dots behind the respective input field.

5. To insert the module call with comment types into the program, confirm with “OK”. The module call will then be inserted.

## Clipboard

The editor supports the Windows clipboard for copying text. Program text can be copied or moved between other Windows applications.

- To select the text section for the clipboard function, hold down the mouse button and move the pointer over the text or hold down the SHIFT key and move the paste position using the direction keys.
- All the text can be selected using the [Edit] [Select All] menu command.
- To copy the selection to the clipboard, use the [Edit] [Copy] command.
- To copy the selection to the clipboard and remove from the file at the same time, use the [Edit] [Cut] command.
- To copy the content of the clipboard to a current position, use the [Edit] [Paste] command.



The content of the clipboard is pasted and is also retained. The [Copy], [Cut] and [Paste] commands also appear in the editor context menu.

Use the DEL key or the [Delete] menu command if you wish to delete the current selection without copying it to the clipboard.



#### Find and replace

Find a specific string as follows:



1. Place the cursor at the start of the section to be searched and select [Edit] [Find...]. The “Find” dialog will open.
2. Enter the text to be searched and set the search options.
3. To start the search, click on the “Find next” button. The dialog remains open. Found text is highlighted.

To continue searching, click on “Find next” again.

4. To close the dialog, click on “Cancel”.

Replace a specific string as follows:

1. Place the cursor at the start of the section to be searched and select [Edit] [Replace...]. The “Replace” dialog will open.
2. Enter the text you are looking for in the “Searching for” field and the new text in the “Replace with” field and set the search options.
3. To start the search, click on the “Find next” button. The dialog remains open. Found text is highlighted.

To replace individual sections of text, click on “Replace”.

If all the found text is to be replaced at one, click on “Replace all”.

4. To close the dialog, click on “Cancel”.

### 3. Programming in statement list



If after closing the dialog you want to find the same text again, select [Edit] [Find next] or press the F3 key. The program will search for the previous found text without opening the dialog.



#### Global find

The [Global Find...] command searches for text or operands in all programs of the entire project. Only the program source texts are searched.



The automatically-generated comments are not part of the program source text.

The “Find operands only” option finds symbolic and absolute operands, regardless of whether the search item was the symbolic or the absolute designation.

The “Only loaded programs” option searches through only the programs selected for loading.

#### Undo changes

You can undo program changes one-by-one.



- To undo a change, select [Edit] [Undo]. Repeat the command if you wish to undo further changes.



- To redo change, select [Edit] [Redo].
- To undo all changes made since the last save, close the program without saving it.



#### **Note**

Changes are automatically saved without a prompt if you:

- change the program properties
- compile the program or the project.

### 3. Programming in statement list

#### Creating backup copy

- If you wish to create a copy of the program, select [Program] [Save As...]. The same dialog used to create a new program will open. The operation neither deletes nor saves the original program.

#### Printing

- In order to print the current status of the program source in the active editor window, select [Program] [Print...].

First answer the prompt asking whether the program is to be saved. The standard dialog for printer selection will then appear.

3.3 Structure of the STL programs

STL sentence                      The STL sentence forms the bottom level of a STL program. Each STL sentence may comprise a condition part and must comprise an executive run part.

Sentence part	Brief description
Condition part	Contains the logical and/or arithmetical links. The condition part always starts with IF. If the programmed links deliver the result “true”, the instructions in the executive part of the sentence are executed.
Executive part	Contains the actions to be executed. The executive part begins with THEN and is run only if the condition part delivers the result “true”.

Tab. 3/2:     Condition and executive part

The short program below shows an example:

Example

```
STEP<label>
IF          SWITCH1      "Condition part
  AND      SWITCH2      "...
THEN SET    LAMP1        "Executive part
OTHRW SET   HORN         "...
IF          "Condition part
THEN      ...            "Executive part
```

Each program line of a sentence comprises:

- a STL keyword (e.g. IF, AND, THEN, SET, OTHRW)
- an operand, e.g. input, output etc. (SWITCH1 and HORN are symbolic operands).

A complete sentence contains an IF part, a THEN part and possibly an OTHRW part.

### 3. Programming in statement list

STL programs can be broken down into steps using the step instruction. These programs are then referred to as step programs. Programs without step structure are referred to as parallel logic programs. The following table provides an overview of the possible structures:

Structure	Brief description
Step program	Program that is broken down into sections (STEPS). Up to 255 steps (1...255) are possible. If the program cannot move onto the next step, the instructions for the current step are executed cyclically (stepwise program execution).
Parallel logic program	Program consisting of pure sentences without step structure. The entire program is run cyclically.
Executive part	Program that consists only of an executive part and contains no condition part.

Tab. 3/3: Possible structures of an STL program

#### 3.3.1 Step program

You can mark each step with a symbolic step label. The jump instruction (JMP TO step label) enables program branching.

A step consists of one or several sentences. The first sentence in a step can be an incomplete sentence. This would be a pure executive part (THEN). This THEN part is always executed without input condition.

The program sequence is stepwise. The program only moves forwards to the next step if in the last sentence of the current step a THEN or OTHRW instruction was executed. If the program cannot move onto the next step, the instructions for the current step (from the start) are executed cyclically.

The STL does not employ edge detection to evaluate the conditions. Subject to the programmed condition being fulfilled, the executive part is run from the start each time the step is executed.

### 3. Programming in statement list

Example of a simple step program:

#### Example

```
STEP label1
  IF                                I1.0
  THEN SET                          F1.5
  OTHRW RESET                       F1.5

STEP label2
  THEN RESET                        F0.0
  IF                                F1.5
  THEN SET                           O0.7
        SET                          F0.0
  OTHRW SET                           O0.0
  JMP TO label1

STEP label3
  IF                                F0.0
  AND                                I0.0
  THEN SET                           O0.4

STEP label4
...
```

### 3. Programming in statement list

#### 3.3.2 Parallel logic program

A parallel logic program consists of pure sentences, i.e. it is programmed without step labels. Otherwise, a parallel logic program is identical to a step from a step program. It is not possible in this case to set-up branches.

The first sentence in a parallel logic program can be an incomplete sentence. All subsequent sentences must be created as complete sentences.

A parallel logic program is (like a LDR program) run cyclically until it is reset. If the program is to be run only once, you must reset it in the last sentence. Example of a parallel logic program (in this case P1), that is run only once:

##### Example

```
THEN   RESET           F0.0
IF
THEN   SET              O0.7
IF
THEN   SET              O1.7
OTHRW  SET              F0.0
        RESET          O1.7
...
...
IF
AND
THEN   SET              O1.0
        RESET          P1
OTHRW  RESET           P1
```

### 3. Programming in statement list

#### 3.3.3 Executive part

In principle, an executive part is structured as an incomplete sentence of the parallel logic program. It does not begin with THEN. In this case, all entered instructions are executed without input conditions. Program branches are not possible. If you enter an IF part underneath, an error message will appear.

##### Example

```
SET          F0.0
RESET        O1.0
LOAD         V50
AFTER        TW7
SET          T7
CMP2
...
```



## 3.4 Brief description of the STL instructions



Further information on the operations in the LDR and STL lists appears in Section 3.3 and Appendix A.

### STL structure

#### STEP

STEP instruction

The STEP instruction enables step programs to be created (see also Section 3.3.1). A maximum of 255 steps per program are possible. The STEP instruction can follow a step label. This may consist of max. 8 characters, i.e. one line. Destination steps for jumps must have a step label.

#### Example

```
STEP setup
...
...
...
THEN JMP TO setup
```



During compilation of the program, the steps are renumbered internally starting from 1.

The program only moves forwards from one step to the next if in the last sentence of the step a THEN or OTHRW instruction was executed.

Step programs do not execute automatic re-runs after the last program step is processed. To re-run the step sequence, the program must jump to the first step.

3. Programming in statement list

**IF**

IF instruction

IF starts a condition part. In the condition part, operands can be requested and linked to form logical and arithmetical expressions. The result of this linking represents the condition for further processing.

**Example**

```
IF          I1.0      " IF 1 signal to I1.0
AND    N      I1.1      " AND 0 signal to I1.1
...
```

**THEN**

THEN instruction

THEN starts the executive part. The following instructions are executed if the condition is true.

**Example**

```
THEN  LOAD      V100
      TO        TP7
...
```

**OTHRW**

OTHRW instruction

OTHRW starts a second alternative executive part. This is executed if the condition part of the step is incorrect and the THEN part is therefore not executed.

**Example**

```
...
THEN  SET      O1.0
OTHRW RESET    O1.0
```



**Condition part**

In the condition part, you can create complex conditions.

You can use the logical links indicated below to link single-bit or multi-bit operands in the condition part (see also Section A.2.5).

**AND**

**AND links**

Logical AND link. The condition is satisfied when all AND linked input conditions are true.

**Example**

IF		I1.0
	AND	I1.1
THEN	SET	O1.0
OTHRW	SET	O1.7

**OR**

**OR link**

Logical OR link. The condition is satisfied if at least one of the input conditions is true.

**Example**

IF		I1.0
	OR	I1.1
	OR	I1.7
THEN	SET	O1.0
OTHRW	SET	O1.7

**EXOR**

**EXCLUSIVE OR link**

Logical EXOR link. This enables precisely two input conditions to be linked together. The condition is satisfied if precisely one of the input conditions is true.

**Example**

IF		I1.0
	EXOR	I1.1
THEN	SET	O1.0
OTHRW	SET	O1.7

**NOP**

**No operation**

NOP denotes No Operation. Use this instruction if you want to execute without an input condition.

**Example**

IF		NOP
THEN	SET	F1.0

**N**

**Negation**

N denotes Negation. It enables you to invert an input condition. In our example, a jump to the Init step occurs if output O1.0 reports a 0 signal.

**Example**

IF		N	O1.0
THEN	JMP TO	Init	

#### Executive part

#### SET

Setting operand logically to 1 (saving)

The SET operation command sets the indicated operand logically to 1 (true).

#### RESET

Resetting operand logically to 0 (saving)

RESET is the opposite to SET. The indicated operand is reset locally to 0 (false).

#### LOAD

Load value to the accumulator

The LOAD instruction loads a value for a single or multi-bit operand to the single-bit or multi-bit accumulator. This usually follows the word TO. The value is then transferred to the target operands.



Source and target operand must be of the same type (either single or multi-bit operand).

#### Example

THEN	LOAD	I0.1	" Single-bit
	TO	F0.1	" Single-bit
	LOAD	V500	" Multi-bit
	TO	TP31	" Multi-bit

#### TO

Transferring value to an operand

The TO instruction loads the content of a single or multi-bit accumulator to a target operand.

### 3. Programming in statement list

#### Example

```
THEN  LOAD      V100    " Load 100
      TO        R6      " to Index 6
      LOAD      F0.1    " Load F0.1
      TO        O0.1    "
```



You can use the logical links indicated below to link multi-bit operands bitwise in the condition part (see also Section A.2.5).

#### AND

Logical AND link (bitwise)

#### OR

Logical OR link (bitwise)

#### EXOR

Logical EXOR link (bitwise)

#### JMP TO

Jump to

The JMP TO instruction executes a jump to a specific step.

#### Example

```
STEP label
IF                                I1.0
THEN SET                          O1.0
      JMP TO Start
...
...

STEP Start
...
```

The program jumps to the step start and is continued.



If you want to prevent the instruction for the current step from being further executed in this cycle, insert a jump at the current step.

#### Special functions

##### SWAP

##### Swap high and low bytes

The SWAP instruction swaps the high-value byte for the low-value byte in the multi-bit accumulator.

##### Example

THEN	LOAD	V\$55AA
	TO	OW0
	SWAP	
	TO	OW1

After OW0 comes \$55AA, after OW1, however, \$AA55 is loaded.

##### SHIFT

##### Shifting single-bit operand

The SHIFT instruction swaps the indicated single-bit operand for the value in the single-bit accumulator. This command can be used to create various lengths of shift register, longer or shorter than the 16-bit word.

First the multi-bit must be loaded and then any SHIFT instruction can be executed.

##### Example

```
STEP 10
  IF      I1.0      "
  THEN LOAD I1.1    "
        SHIFT O1.1  " swap F0.0
        SHIFT O1.2  " swap O1.1 <-> O1.2
        SHIFT O1.1  " swap O1.2 <-> O1.3
        SHIFT O1.4  " swap O1.3 <-> O1.4
STEP 20
  IF      N          I1.0      " wait until input
                                " is deactivated
  THEN JMP TO 10      " repeat
```

**SHL**

**Shift to left**

Shift to left. This instruction enables you to shift the content of the multi-bit accumulator by one bit space to the left. The empty space to the right is occupied by a zero. This denotes multiplication by 2. If you select the STL instruction three times in succession, it denotes multiplication by 2x2x2, i.e. by 8.

**Example**

THEN	LOAD	V16
	SHL	
	TO	R7

The content of R7 is then 32.

**SHR**

**Shift to right**

Shift to right. This instruction enables you to shift the content of the multi-bit accumulator by one bit space to the right. The empty space to the left is occupied by a zero. This denotes division by 2. As with SHL, each multiple shift causes a division by 2.

**Example**

THEN	LOAD	V16
	SHR	
	TO	R7

The content of R7 is then 8.

**ROL**

**Rotate to left**

This instruction has the same effect as SHL except that the highest-value bit is pushed completely to the left out of the accumulator and reaccepted as an overflow to the right as the lowest-value bit.



### 3. Programming in statement list

#### ROR

#### Rotate to right

As with SHR, the bits of the multi-bit accumulator are moved to the right. In this case, however, the right bit is pushed out of the accumulator and accepted as the highest-value bit.

#### BID

#### Converting binary to decimal display

The BID instruction converts the content of the multi-bit accumulator from binary into decimal display. The BCD code enables you for example to control LEDs in displays.

#### Example

THEN	LOAD	IW0
	BID	
	TO	OW7

DEB

Converting decimal to binary display

The DEB instruction converts the content of the multi-bit accumulator from decimal to binary code. This is practical, for example, if you have connected decimal switches to the input of your controller, the switching conditions for which have been accepted by the input word and are then to be processed in a counter.

Example

THEN	LOAD	IW7
	DEB	
	TO	CW7

INV

Inversion – 1’s complement

The INV command inverts each bit of a multi-bit accumulator to create the 1’s complement.

Example

THEN	LOAD	OW1
	INV	
	AND	IW1
	TO	OW1

CPL

2’s complement

The CPL command creates the 2’s complement for the multi-bit accumulator. In this case, all bits are negated (1’s complement) and then added by 1. In the case of whole numbers with sign, this is equivalent to multiplying by -1.

### 3. Programming in statement list

#### Example

```
IF                ( R32
                  < V0 )
THEN  LOAD        R32
      CPL
      TO          R22
```

#### Arithmetical instructions

##### INC

##### Increment

The increment instruction (count upwards) increases the value of the following multi-bit operand by 1. Unlike other arithmetical instructions, the multi-bit operand to be incremented does not need to be loaded to the multi-bit accumulator beforehand.

The INC instruction can be used to increment all (writable) multi-bit operands, although is usually used in combination with counters.

#### Example

```
IF                I1.3
THEN  INC         R9
```

##### DEC

##### Decrement

The decrement instruction (count downwards) decreases the value of the following multi-bit operand by 1. Unlike other arithmetical instructions, the multi-bit operand to be decremented does not need to be loaded to the multi-bit accumulator beforehand.

The DEC instruction can be used to decrement all (writable) multi-bit operands, although is usually used in combination with counters.

3. Programming in statement list

Example

IF			I 2 . 2
	AND	N	I 3 . 6
THEN	DEC		R 9

**+, -, \*, /, <, <=, =, >=, >, <>**

Basic calculation methods and comparison

In addition to previously stated commands, the following mathematical operations are available to you:

Operation	Description
()	Brackets denote priority processing
+	Addition
–	Subtraction
*	Multiplication
/	Division
<	Comparison: smaller than
<=	Comparison: smaller than or equal to
=	Comparison: equal
>=	Comparison: greater than or equal to
>	Comparison: greater than
<>	Comparison: unequal

Tab. 3/4: Basic calculation methods and comparison

These operations enable you to program calculations and comparisons.

### 3. Programming in statement list

#### Example

```
IF          (   FW0
            =   V1234   )
            AND
            (<>   R1
            V0    )
THEN...
```



With such instructions, respect whether and for which expressions you have to use brackets. The logical set-up can easily be damaged by incorrect or missing brackets.

#### Module calls

##### CFM

#### Calling function module

The CFM command is used to call a function module. Function modules are sub-programs that can be created in STL, LDR and C. They **can not**, however, contain steps. Following a function module call, there is **no** task change.

When called, function modules can be provided with input parameters and deliver values in feedback parameters to the selecting program. The parameters are transferred to the local program function units FU32 to FU38.



A precise specification of the parameter transfer for pre-prepared function modules appears in the documentation on the relevant module (see also description of the WITH command). Further information on modules can be found in Appendix A.5.

CMP

Calling program module

The CMP command is used to call a program module. Program modules are also sub-programs that can be created in STL, LDR and C. They **can**, however, contain steps. After a program module is selected, a **task change always takes place** to enable the program module steps to be processed.

When called, program modules can also be provided with transfer parameters and deliver values in feedback parameters to the selecting program. The parameters are transferred to the local program function units FU32 to FU38 (see also description of the WITH command).

The CMP command must not be used in program modules. Further information on modules can be found in Appendix A.5.

WITH

WITH instruction

The WITH instruction is used for transferring parameters to program and function modules. The parameters are transferred to the local program function units FU32 to FU38.

Example

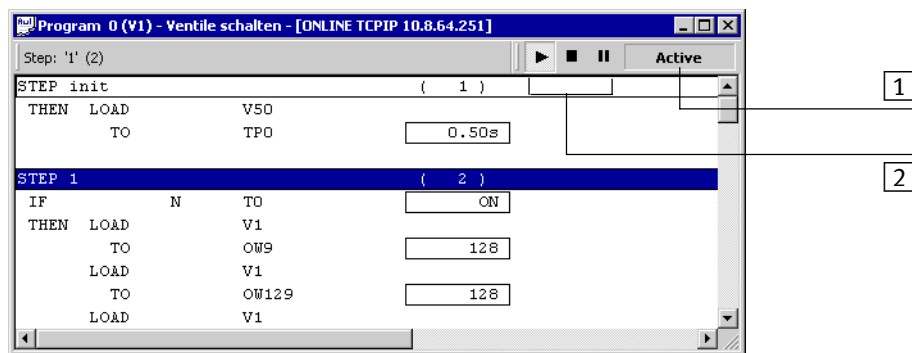
```
IF      I1.2
THEN   CFM 0
      WITH V4    " Realtime clock:
      WITH V30   " Set clock
      WITH V0    " Hour, 0..23, (FU32)
      WITH V0    " Minute, 0..59, (FU33)
      WITH V0    " Seconds, 0..59, (FU34)
      WITH V0    " Hundredths of secs, 0..99,
                  " (FU35)
```

## 3.5 STL online display

The online display of the STL is an effective trouble shooting tool. While the controller program is running, status information and operand values are displayed and can be changed.

Start the online display as follows:

1. Open the required program.
2. Select [Online] [Online] or the [Online] command from the context menu of the STL editor. The online display will then be started.



**1** Displaying the program status

**2** Buttons for program control (start, stop, halt)

Fig. 3/8: Online display

The active step is highlighted. The name and number of the active step, the program status and any runtime errors are displayed in the status line (below the title bar). You can use the buttons in the status line to start, stop or halt the displayed program.



You can use the online control panel to start, stop or halt the entire project.

### 3. Programming in statement list

The current values of the operands are displayed to the right next to the program source. If the mouse pointer is over an operand value, the corresponding identifier will be displayed.

- If you want to view a specific section of the program, use the window scroll bars or select the [Goto...] command from the context menu. This command enables you to jump straight to the required program section by entering a step number.

#### 3.5.1 Functions of the STL online display

The context menu of the STL online display offers various online commands. These are explained below.

##### Update speed and display format

As with the online display (see Section 2.9.4), you can change the following for the display:

- the update speed
- for multi-bit values, the display format (decimal with prefix, decimal without prefix or hexadecimal).

Further information can be found in section 2.9.4.

##### Changing or forcing operand values

As with the online display, you can also carry out the following actions (see Section 2.9.4):

- Forcing inputs and outputs
- Changing operand values





### 3. Programming in statement list

However, when changing single-bit operands, ensure that the status of single-bit operands is shown in the STL online display by entering “On” and “Off”.

- “On” denotes 1 signal.
- “Off” denotes 0 signal.

Click on the value (“ON” or “OFF”) to change the signal status of the operand.



Further information on forcing and changing multi-bit operands can be found in Section 2.9.4.

#### Executing active step (for step programs only)

- To execute the active step of the program shown in the online display, select the [Execute active Step] command from the context menu. The step will then be executed. When the step is exited, the program is halted again.

#### Executing a cycle

- To execute only one step of the program shown in the online display, select the [Execute one Cycle] command from the context menu. The program will then run a PLC cycle.

#### Breakpoints



##### **Note**

To set breakpoints, a runtime main program Version 2.26 or higher is required. The CI terminal shows you, for example, the version in use (see also Fig. 2/65). With older versions, the functions described below are not available.

3. Programming in statement list



For test purpose, you can set breakpoints in the online display. The program is then halted and is not continued until the next start command.

A distinction is made between the following breakpoint types:

Breakpoint type	Designation	Description
Temporary (Only once)	Yellow bar	The program is broken at the defined point and the breakpoint is then automatically deleted.
Permanent (Always)	Red bar	The program is broken at the defined point without the breakpoint being automatically deleted.

Tab. 3/5: Breakpoint types (temporary or permanent)

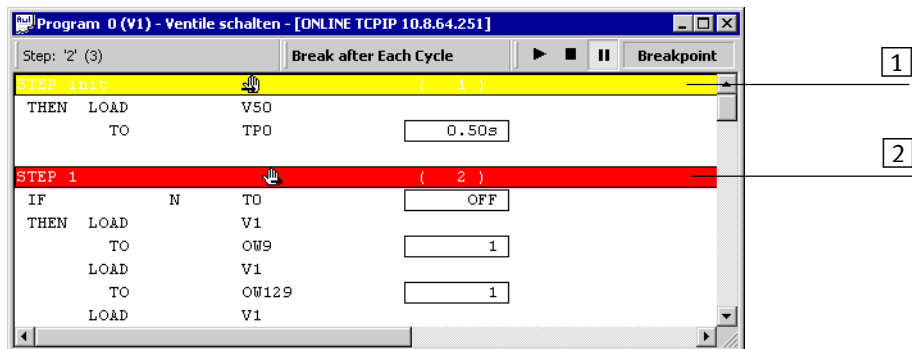
STL step programs support the following breakpoint variants:

Breakpoint variants			Description
Temporary (yellow)	Permanent (red)		
Break on Enter			The program is broken <b>before</b> the selected step is executed.
Break on Leave			The program is broken <b>after</b> the selected step is executed.
Break after Next Cycle	Break after Each Cycle		The program is broken after the cycle is executed.

Tab. 3/6: Breakpoint variants

### 3. Programming in statement list

The following diagram shows an example of breakpoints in the STL online display:



- [1] Temporary breakpoint (yellow) [2] Permanent breakpoint (red)

Fig. 3/9: Showing breakpoints in the STL online display

Insert breakpoints as follows:

1. Open the desired program and activate the online display.
2. Select the [Breakpoints...] command from the context menu. The following window will then appear:

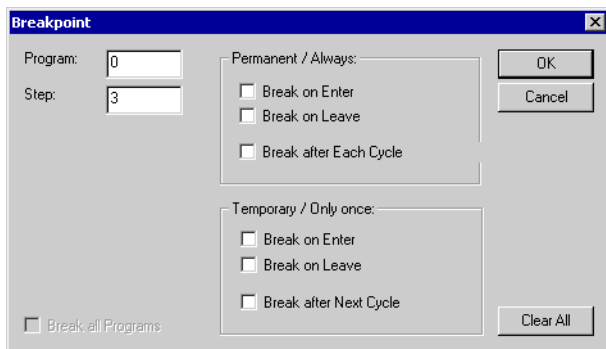


Fig. 3/10: Breakpoint

### 3. Programming in statement list

3. Enter the required program and step number in the program field and step field respectively.
4. Select the required breakpoint (see Tab. 3/5 and Tab. 3/6).



Up to 16 breakpoints are permitted for the project. You may, if necessary, also specify several breakpoint variants for the selected step. If both permanent and temporary breakpoints exist at a location, only the designation for the permanent breakpoints (red bar) is shown in the STL online display. The [Breakpoints...] command in the [Online] menu enables you to view a list of all breakpoints in the project.

5. If all programs are to be halted when the breakpoint is reached, select the “Break all Programs” option.



The project can be continued using the Start command in the online control panel.

6. To conclude, confirm with OK. The breakpoints are then transferred and become valid.

### 3. Programming in statement list

#### List of all breakpoints

All breakpoints set in the project can be displayed and edited in an overview.

Open the breakpoint list as follows:

- Select the command [Breakpoints...] command from the [Online] menu. The following dialog will then open.

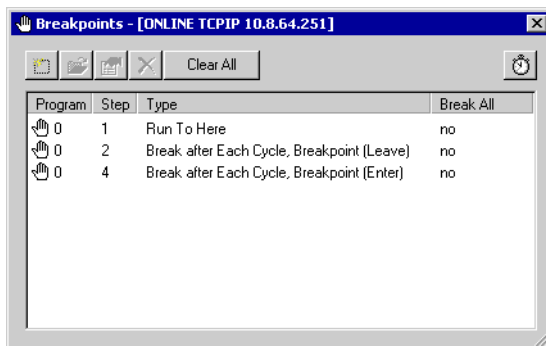






Fig. 3/11: Breakpoint list



If several breakpoints are defined at one location, they are listed successively in the Type column.

### 3. Programming in statement list

The following operations are available for editing the breakpoints:

Icon	Operation
	Define new breakpoint
	Open program at selected breakpoint
	Display/change breakpoint properties
	Delete selected breakpoint
Clear All	Delete all breakpoints in the project

Tab. 3/7: Operation in “Breakpoints” dialog window

#### 3.5.2 Deactivating STL online display

- Select [Online] [Editor] or the [Editor] command from the context menu. The STL editor will then be opened.

# **Programming in ladder diagram**

## **Chapter 4**

Contents

**4. Programming in ladder diagram ..... 4-1**

4.1 Fundamental principles for LDR programming ..... 4-4

4.2 The ladder diagram editor ..... 4-6

4.2.1 LDR editor preferences ..... 4-7

4.2.2 Tagging a selection in the LDR editor ..... 4-8

4.2.3 Functions of the LDR editor ..... 4-10

4.3 LDR symbols ..... 4-19

4.3.1 Symbol in the condition part ..... 4-19

4.3.2 Symbol in executive part ..... 4-21

4.4 LDR online display ..... 4-27

4.4.1 Functions of the LDR online display ..... 4-28

4.4.2 Deactivating LDR online display ..... 4-30



## 4. Programming in ladder diagram

Contents of this chapter	<p>This chapter describes the functions of the LDR editor and the options of the LDR online display. The LDR editor is used to create your user programs. The LDR online display enables you to display status information and also display and change operand values while the controller program is running.</p> <p>It also contains a brief description of the LDR symbols, which provide you with an initial overview of the programs in the ladder diagram.</p>
Further information	<p>A detailed description of STL and LDR operations appear in Appendix A.</p>

### 4.1 Fundamental principles for LDR programming

Ladder diagram – LDR for short – is a graphic-based programming language developed from the circuit diagram. The diagram of a LDR program is therefore similar to the diagram of a circuit diagram – in relation to the diagram of logical links. However, for the LDR diagram, new symbols have been introduced for contacts and coils that are better suited for displaying on a monitor.



Due to the similarities with circuit diagrams, the LDR diagram program is frequently preferred by developers who are familiar with relay technology. If a circuit diagram already exists for a control task, it can usually be transferred to a LDR program.

A LDR diagram is based on two vertical lines. In the transfer sense, the left line is linked to the voltage source and the right is earthed. Between them, the LDR diagram is compiled in the form of horizontally arranged rungs with contacts, coils and other LDR symbols.

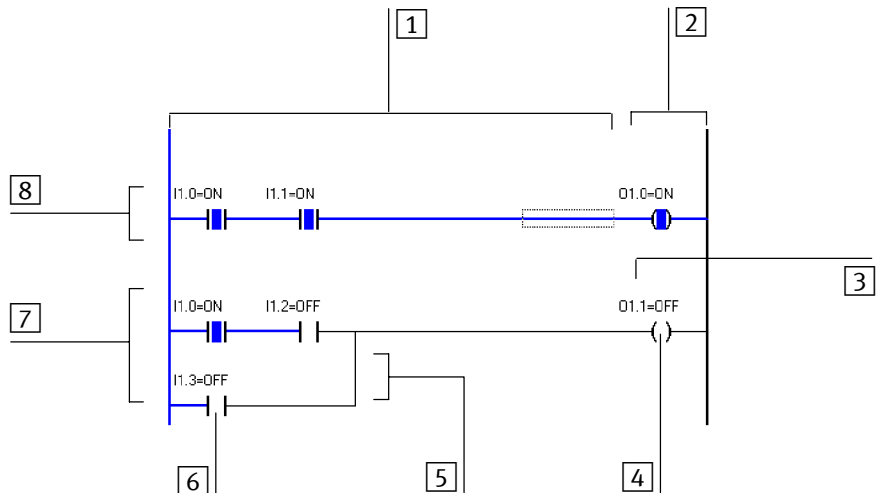
Rungs consist of a condition part and an executive part. The left side of a rung represents the condition part, which contains the logical and/or arithmetical links, e.g. in the form of contacts and parallel branches. The right side of a rung represents the executive part. This is where the action to be executed is programmed, e.g. in the form of coils. A rung in the LDR therefore usually reads from left to right.

As an example, the diagram below shows a small section of a LDR program in the online display.



When the FST software is in online mode, contacts and coils and all lines that report 1 signal are highlighted in blue. Operands that report 1 signal are tagged with “ON”, Operands that report 0 signal with “OFF”.

#### 4. Programming in ladder diagram



- |                  |   |
|------------------|---|
| 1 Condition part | 5 Parallel branch in the condition part |
| 2 Executive part | 6 Contact symbols                       |
| 3 Operand        | 7 Rung 2                                |
| 4 Coil symbols   | 8 Rung 1                                |

Fig. 4/1: Extract from a LDR program (online display)

During programming each symbol is assigned an operand, e.g. a real, existing input or output of the PLC/IPC.

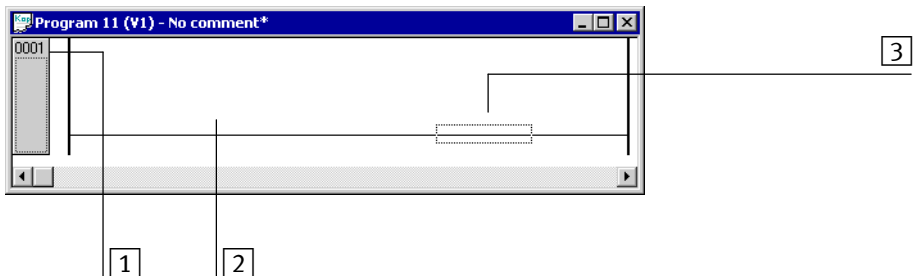
The symbols in the condition part (e.g. contacts) transfer signals “1” or “0” and/or the Boolean values “true” or “false” in a right hand direction towards the executive part. Signals are AND-linked through the successive switching of contacts and OR-linked through parallel switching.

### 4.2 The ladder diagram editor

The LDR editor offers the following functions:

- fast selection of all LDR symbols via shortcut bar or selection of all LDR symbols via menu commands
- automatic allocation list entry and insertion of operands and allocation list comments through selection from the list
- dialog-controlled insertion of module calls
- editing functions such as copying, find/replace, undo etc.
- various configuration options for the program display, e.g. font style, font size etc.

In order to create a LDR program, you must first build a project (see Section 2.2.1). Any new LDR program you insert or existing one you open will be shown in the LDR editor window. With new LDR programs, the rung will appear empty.



**1** Rung number

**3** Selection for connection line

**2** Empty rung

Fig. 4/2: LDR editor with empty rung

## 4. Programming in ladder diagram

### 4.2.1 LDR editor preferences

The [Extras] [Preferences...] command enables you to set the following preferences for the LDR editor:

“General” tab (preferences valid for STL and LDR editors):

- automatic activation/de-activation of an allocation list item
- display symbol/operand with allocation list comment

LDR tab:

- maximum number of rung comment lines (see also Fig. 4/7)
- maximum number of operand comment lines (see also Fig. 4/8)
- Font preferences, e.g. for operands, comments, jump labels etc.

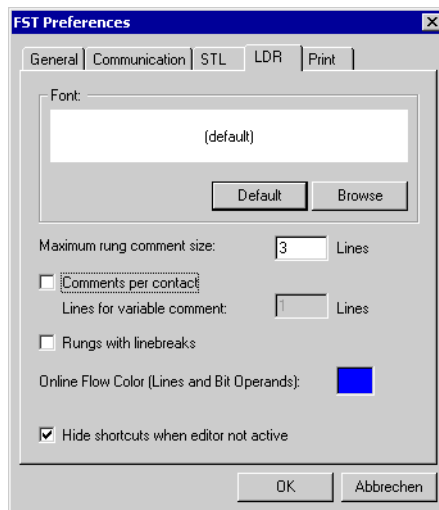


Fig. 4/3: LDR editor preferences

## 4. Programming in ladder diagram

### 4.2.2 Tagging a selection in the LDR editor

A selected graphic element is tagged with a dotted triangle. Selected text is highlighted. The following elements can be selected:

- one or several rungs
- one or several LDR symbols in the condition part
- a LDR symbol in the executive part
- the tie line between the condition part and the executive part (see also Fig. 4/2)
- each text block with a text field.

Examples of selection tagging are shown in the diagram below:

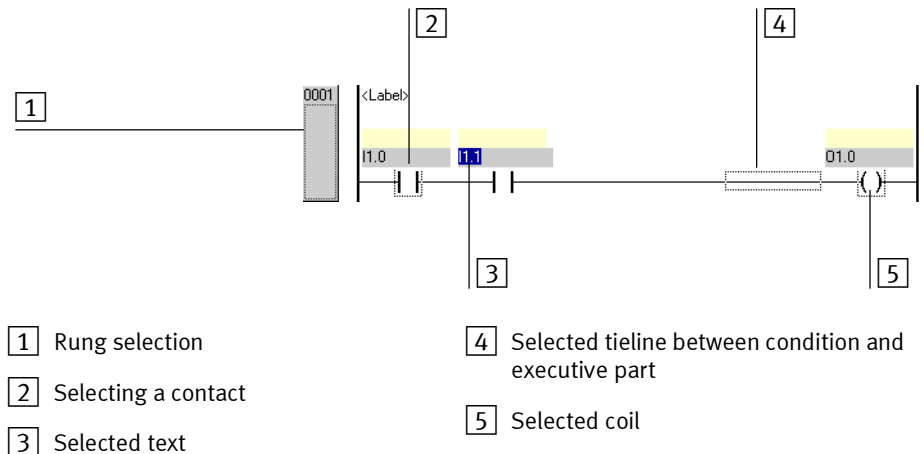


Fig. 4/4: Selection in the LDR editor

## 4. Programming in ladder diagram

### Selecting with the mouse

- Click to select a LDR symbol or a text field.
- To select a rung, click on the field below the rung number.

You can select several rungs or several LDR symbols in the condition part.

- To extend the current selection, hold the SHIFT KEY down and select the last element to which the selection is to be extended.

**1** Extended selection (dotted triangle)

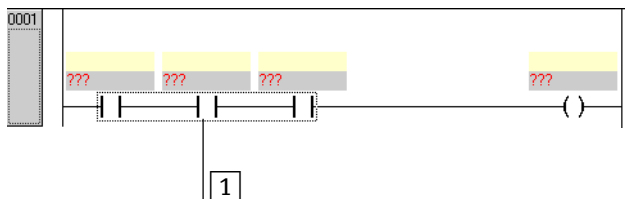


Fig. 4/5: Example of an extended selection in the condition part

### Selecting with the keyboard

Use the arrow keys to select the next element in the direction you wish to go. All elements, including the text fields can be reached by the keyboard. If the tie line between condition part and executive part is selected, you can use the PAGE UP or PAGE DOWN key to move the selection to the preceding or next rung.

You can select several rungs or several LDR symbols in the condition part.

- To extend the current selection, hold the SHIFT KEY down and use the relevant DIRECTION KEY to select the last element to which the selection is to be extended.

4.2.3 Functions of the LDR editor

Inserting LDR elements

The [Insert][LDR Element] menu is provided for inserting new rungs and LDR symbols.

LDR shortcut bar

The LDR shortcut bar offers direct access to all LDR elements.

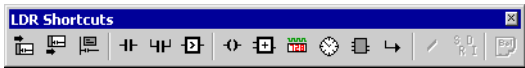





Fig. 4/6: Shortcuts

- To open the shortcut bar, select [View][Shortcuts] or [Shortcuts] from the context menu.













The shortcut bar is automatically hidden if the LDR editor window is no longer the active window. The shortcut bar can be placed in any position as a separate window. It can also be moved with the mouse to below the menu bar or arranged on any window edge.

The table below describes the shortcut bar elements:

Icon	Description	Action
Rung symbols		
	Rung	Insert after the current rung
	Rung	Insert before the current rung
	Comment field for rung	Insert for the current rung



#### 4. Programming in ladder diagram

Icon	Description	Action
Symbols for the condition part		
	Contact (closer)	Insert before current selection
	Contact in parallel branch	Insert parallel to current selection
	Comparison box	Insert before current selection
Symbols for the executive part		
	Coil	Insert after current selection
	Multi-bit operations	
	Counter box	
	Timer box	
	Program or function module	
	Jump	
Other symbols		
	Negation	Modify current selection: – Negation with contacts and coils – S/R with coils – D/I possible with multi-bit operands
	Set/Reset/Decrement/Increment	
	Allocation list	Selecting operand from the allocation list

Tab. 4/1: Symbols on the shortcut bar

## 4. Programming in ladder diagram

Observe the following when inserting LDR symbols:

- Condition symbols are inserted before the current selection in the condition part. If a coil or the tie line between the contacts and the coils is currently selected, the symbol is inserted as the last condition symbol.
- Parallel branches are inserted parallel to current selection.
- Execution symbols are inserted as follows:
  - Jump symbols as last executive symbol.
  - Coils after the last coil
  - Executive boxes after the last executive symbol, if necessary before the jump symbol.

Insert a rung or a LDR symbol into the program as follows:

1. Select the element before or after which you wish to insert the new LDR element.



Only those elements valid for the current selection can be selected.

2. Click on the required symbol in the LDR shortcut. The element will then be inserted.

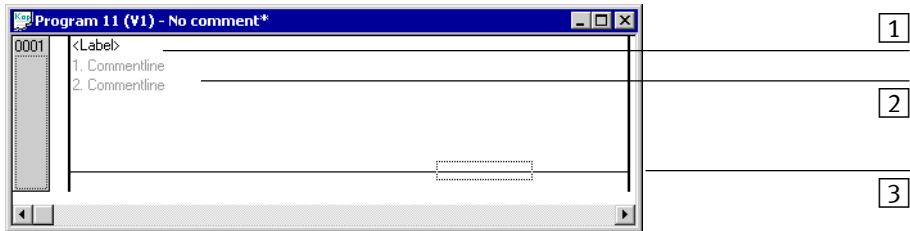
4. Programming in ladder diagram

Rung number, jump label and comment

- Rung number

LDR programs consist of rungs. These are tiled vertically and can be separated by a tie line. Each rung is tagged on the left side with a continuous number. The numbers are assigned automatically.
- Jump labels

For each rung, one jump label can be entered (max. 9 characters or figures). A maximum of 255 jump labels per program are possible. The input field for the jump label is located adjacent to the rung number.



- 1

Input field for a jump label
- 2

Input field for comments
- 3

Tie line

Fig. 4/7: Jump label comment

- Rung comment

For each rung, you can enter a multi-line comment (see Fig. 4/7).



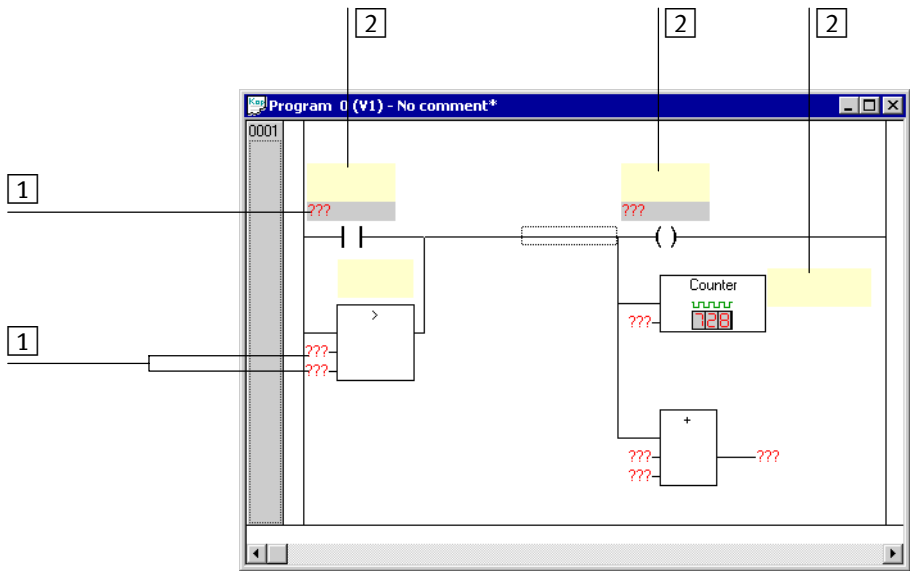
You can set the maximum number of lines for comments using the [Extras] [Preferences...] command in the “LDR” index.

#### 4. Programming in ladder diagram

##### Operand and operand comment

When you insert a new LDR symbol into a rung, e.g. a contact, an input field for the operand is automatically opened. 3 question marks (???) are entered as preset. You can enter the operand directly by overwriting.

Depending on LDR symbol, a comment field is displayed for the operand. An example is shown in the diagram below.



1 Input field for absolute or symbolic operand

2 Input field for operand comment

Fig. 4/8: Input field for operands and operand comments



You can set the maximum number of lines for comments using the [Extras] [Preferences...] command in the "LDR" index. Operands and operand comments can be automatically transferred from the allocation list.



### Automatic allocation list entry

The [Extras] [Preferences...] command enables you to activate or deactivate automatic allocation list entry for the STL and LDR editor together in the “General” tab.

If the function is active and you enter a symbolic or absolute operand into the program, which is already entered in the allocation list with a comment, the comment is transferred from the allocation list into the program. Unless, however, you have entered a comment manually.

When an operand is entered which has not yet been input into the allocation list, you will be offered the dialog to enter it into the allocation list.

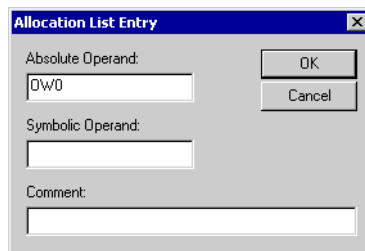


Fig. 4/9: Dialog for automatic allocation entry

Press “OK” to transfer the operand into the program and the allocation list.

- If you do not wish to transfer the operand into the allocation list, click on “Cancel” or press the ESC key. The operand will then be transferred only to the program.

Selecting operand from the allocation list

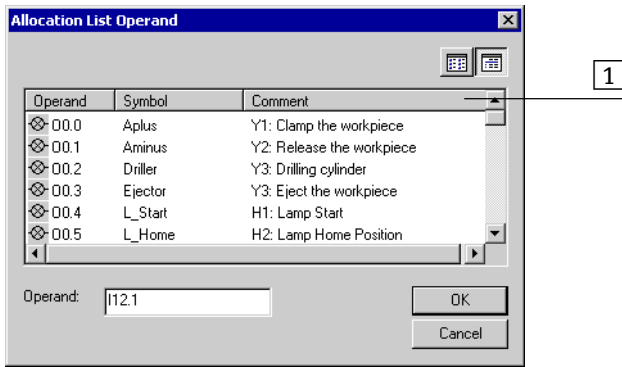
Instead of entering an operand directly into the program, you can select it from the allocation list.

Selecting an operand

Select an operand from the allocation list as follows:



- 1. Select [Insert] [Operand...] or click on “Allocation List Operand” in the shortcut window. The following dialog will then appear.



1 Table header

Fig. 4/10: Selecting from the allocation list

- 2. Select the desired operand and confirm your selection with OK.



You can change the width of the columns in the table header using the mouse. Each column of the window is sorted alphabetically by clicking on the column header. Clicking again reverses the sorting sequence.

### Clipboard

The editor supports the Windows clipboard for copying text. An internal clipboard is used to copy graphic elements. copy or move text or LDR symbols or entire rungs within the FST software. Text can also be copied or moved between other Windows applications.

- To copy the selection to the clipboard, use the [Edit] [Copy] command.
- To copy the selection to the clipboard and remove from the file at the same time, use the [Edit] [Cut] command.
- To insert the content of the clipboard, select [Edit] [Paste] command.



The content of the clipboard is pasted and is also retained. The [Copy], [Cut] and [Paste] commands also appear in the editor context menu.

Use the DEL key or the [Delete] menu command if you wish to delete the current selection without copying it to the clipboard.

### Find/Replace/Global find

As when working with the STL editor, you can also find and replace strings in the LDR editor, e.g. operands and comments. Further information can be found in “Find...” and “Replace” and “Global Find...” in Section 3.2.2.

## 4. Programming in ladder diagram

### Undo changes

You can undo program changes one-by-one.



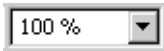
- To undo a change, select [Edit] [Undo]. Repeat the command if you wish to undo further changes.
- To redo change, select [Edit] [Redo].
- To undo all changes made since the last save, close the program without saving it.



#### **Note**

Compiling a project or running a syntax check on the program saves all changes.

### Zoom function



A picklist in the symbol bar enables you to control the size in which a LDR program is to appear on the screen. Possible values range between 25 % and 400 %. Individual values of between 10 % and 500 % can be entered manually. The default value for all objects is 100 %.



The zoom function is available only in the LDR Editor. The set zoom level is saved in the project as a property. Changes in size do not affect the printout.



### 4.3 LDR symbols

The following sections provide an overview of the different LDR symbols and/or the corresponding LDR operations.



Further information on the operations in the LDR and STL lists appears in the Appendix A.

#### 4.3.1 Symbol in the condition part

##### Contacts

Contacts request signals from inputs, outputs and other single-bit operands. A distinction is made between closer and opener. In the FST software, closers are shown as two vertical lines in parallel (| |). Openers (|/|) are generated by negating (/) a closer. The single-bit operand to which the request is to be made is entered directly above the contact symbol.

Contact	Operation	Description
	Request for 1 signal (Closer)	A present 1 signal is transferred to the right if the operand reports 1 signal. Otherwise, a 0 signal is transferred.
	Request for 0 signal (Opener)	A present 1 signal is transferred to the right if the operand reports 0 signal. Otherwise, a 0 signal is transferred.

Tab. 4/2: Contacts

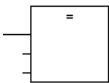


Signals are AND-linked through the successive switching of signals and OR-linked through parallel switching (see Section A.1.5).

4. Programming in ladder diagram

Comparison boxes (symbol in the condition part)

A comparison box enables multi-bit operands to be compared. A comparison enables simple analogy values to be processed, step chains to be set-up, counters to be evaluated repeatedly etc. The following operations are supported:

Icon	Operator	Operation	Description
	=	Equal to	A present 1 signal is transferred to the right if the comparison is satisfied. Otherwise, a 0 signal is transferred.
	>	Greater than	
	<	Less than	
	>=	Greater than or equal to	
	<=	Less than or equal to	
	<>	Unequal to	

Tab. 4/3: Comparison box

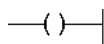
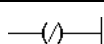
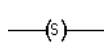
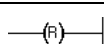
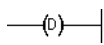
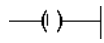
The 2 multi-bit operands to be compared – top Operand 1, bottom Operand 2 are entered into the input side of the comparison box (left). The operation is entered into the comparison box. Equal to (=) is the default sign.

## 4. Programming in ladder diagram

### 4.3.2 Symbol in executive part

#### Coil symbols

Coils represent the outputs of the controller via which the actuators are addressed following processing of the input signals. When programming a PLC/IPC, you can influence all single-bit and multi-bit operands via the corresponding coil symbols. Coils are shown in brackets (). The selected operation is displayed in the bracket (see Tab. 4/4). The operand to which the operation is to relate is entered directly above the coil. The following operations are available:

Coil	Operation	Description
Single-bit operation		
	ASSIGNMENT (Non-saving)	Transfers the logical status and/or the negated logical status of the condition directly to the coil and/or single-bit operand. The operand also follows each change of the logical status in the condition part.
	NEGATED ASSIGNMENT (Non-saving)	
	SET (Saving)	With an rising edge on the condition part, the operand is set with saving (1 signal) and/or reset with saving (0 signal). Otherwise, the status of the operand is not influenced.
	RESET (Saving)	
Multi-bit operation		
	DECREMENT (Decrement)	With a rising edge on the condition part, the value of the operand is decreased by 1 (implicit edge detection).
	INCREMENT (Increment)	With a rising edge on the condition part, the value of the operand is increased by 1 (implicit edge detection).

Tab. 4/4: Coils

### Boxes

Operations in the executive part besides those shown by coils (see Section 4.3.2) are shown by a box. The following operations are possible:

- Multi-bit operations
- Counter
- Timer
- Module call.



All actions defined by a BOX in the executive part are run with an rising edge of the link result.

### Multi-bit operations

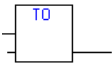
Words are processed by multi-bit operations. The operands in Festo PLC/IPCs are organised into 16 bit wide words. Each word consists of two bytes (8 bits per byte). The multi-bit operations can be sub-divided into:

- Multi-bit operations with 2 operands
- Multi-bit operations with 3 operands

4. Programming in ladder diagram

Multi-bit operations with 2 operands

In the case of multi-bit operations with 2 operands, the operation is applied to the first operand (left). The result of the operation is stored in the second operand (right).

Icon	Operator	Operation	Brief description
	LOAD TO	Load to	Loads the content of a multi-bit operand to another multi-bit operand
	SWAP	Swaps bytes	Swaps low and high bytes
	SHL	Shift to left	All bits of a multi-bit operand are shifted to the left; right bit becomes 0
	SHR	Shift to right	All bits of a multi-bit operand are shifted to the right; left bit becomes 0
	ROL	Rotate to left	Moves all bits left and accepts the bit pushed out the left as a low-value bit.
	ROR	Rotate to right	Moves all bits right and accepts the bit pushed out the right a a highest-value bit.
	INV	Invert	Inverts all bits
	CPL	Create complement	Reverse sign
	BID	BINARY to BCD	Converts BINARY to BCD display
	DEB	BCD to BINARY	Converts BCD to BINARY display

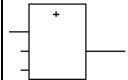
Tab. 4/5: Overview Multi-bit operations with 2 operands

The multi-bit operand to which the operation is to be applied is entered at the input side of the box (left). The operand to which the result of the operation is to be saved is entered at the output side (right). The operation is displayed in the box.

4. Programming in ladder diagram

Multi-bit operations with 3 operands

In the case of multi-bit operations with three operands, the operation is applied to the first two operands. The result of this operation is saved to the third operand. The same operand can be entered as the first, second and third operand.


Icon	Operator	Operation	Brief description
Multi-bit operations with 3 operands			
	AND	AND link	Logical AND linking with words
	OR	OR link	Logical OR linking with words
	EXOR	EXOR link	Logical EXOR linking with words
	*	Multiplication	Multiplying by words
	/	Division	Dividing by words
	+	Addition	Adding words
	-	Subtraction	Subtracting words

Tab. 4/6: Overview Multi-bit operations with 3 operands

The two multi-bit operands to which the operation is to be applied – Operand 1 top, Operand 2 below – are entered at the input side of the box (left). The operand to which the result of the operation is to be saved is entered at the output side (right). The operation is displayed in the box.

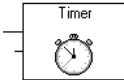
#### 4. Programming in ladder diagram

##### Counter

Icon	Operation	Description
	Counter initialisation	Loads the indicated value into the counter preset

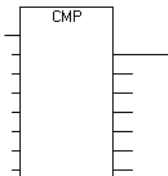
Tab. 4/7: Counter box

##### Timer

Icon	Operation	Description
	Initialises and starts timer	Loads the indicated value into the timer preset and starts the timer

Tab. 4/8: Timer box


##### Module call

Icon	Operation	Description
	Module call (in this case CMP)	Calls the indicated program module (CMP) or function module (CFM) and transfers or accepts the indicated parameter

Tab. 4/9: Module call

4. Programming in ladder diagram

Jump

Icon	Operation	Description
	Jump	Interrupts the program run at the current location and continues it from the indicated jump label in the current program.

Tab. 4/10: Jump



### 4.4 LDR online display

The LDR online display is an effective trouble shooting tool. While the controller program is running, status information and operand values are displayed and can be changed.

Start the online display as follows:

1. Open the required program.
2. Select [Online] [Online] or the [Online] command from the context menu of the LDR editor. The online display will then be started.

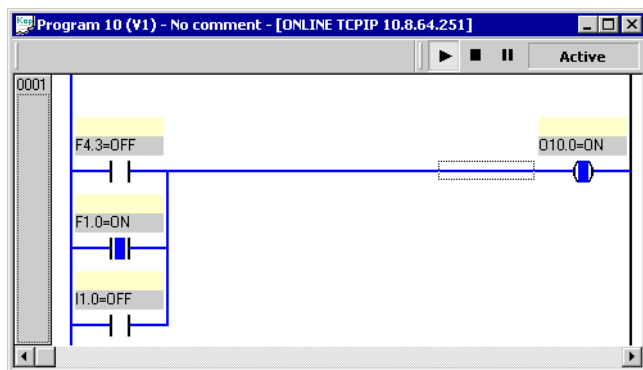


Fig. 4/11: Online display

The program status and any runtime errors are displayed in the status line (below the title bar). You can use the buttons in the status line to start, stop or halt the displayed program.



You can use the online control panel to start, stop or halt the entire project.

Connected paths and condition symbols that are satisfied are highlighted in blue.

## 4. Programming in ladder diagram

The current values of the operands are displayed over the LDR symbols. If the mouse pointer is over an operand value, the corresponding identifier will be displayed.

- If you want to view a specific section of the program, use the window scroll bars or select the [Goto...] command from the context menu. This command enables you to jump straight to the required program section by entering a rung number.

### 4.4.1 Functions of the LDR online display

The context menu of the LDR online display offers various online commands. These are explained below.

#### Update speed and display format

As with the online display (see Section 2.9.4), you can change the following for the display:

- the update speed
- for multi-bit values, the display format (decimal with prefix, decimal without prefix or hexadecimal).

Further information can be found in section 2.9.4.



#### Changing or forcing operand values

As with the online display, you can also carry out the following actions (see Section 2.9.4):

- Forcing inputs and outputs
- Changing operand values

## 4. Programming in ladder diagram

However, when changing single-bit operands, please note that the status of single-bit operands is shown in the LDR online display by “On” and “Off”.

- “On” denotes 1 signal.
- “Off” denotes 0 signal.

Click on the value (“ON” or “OFF”) to change the signal status of the operand.



Further information on forcing and changing multi-bit operands can be found in Section 2.9.4.

### Executing a cycle

- To execute only one step of the program shown in the online display, select the [Execute one Cycle] command from the context menu. The program will then run a PLC cycle.

### Breakpoints



#### **Note**

To set breakpoints, a runtime main program Version 2.26 or higher is required. The CI terminal shows you, for example, the version in use (see also Fig. 2/65). With older versions, the functions described below are not available.

For test purposes, you can set breakpoints in the online display. The program is then halted and is not continued until the next start command.

4. Programming in ladder diagram

A distinction is made between the following breakpoint types:

Breakpoint type	Designation	Description
Temporary (Only once)	With LDR none	The program is halted at the defined point and the break-point is then automatically deleted.
Permanent (Always)	With LDR none	The program is halted at the defined point without the breakpoint being automatically deleted.

Tab. 4/11: Breakpoint types (temporary or permanent)

LDR step programs support the following breakpoint variants:

Breakpoint variants		Description
Temporary	Permanent	
Break after Next Cycle	Break after Each Cycle	The program is halted after the cycle is executed.

Tab. 4/12: Breakpoint variants



Further information on inserting and editing breakpoints can be found in Section 3.5.1.

4.4.2 Deactivating LDR online display

- Select [Online] [Editor] or the [Editor] command from the context menu. The LDR editor will then be opened.

# **Fundamental principles of the FST operating system**

## **Chapter 5**

## Contents

<b>5.</b>	<b>Fundamental principles of the FST operating system</b>	<b>5-1</b>
5.1	Operands (operating resources of the controller)	5-4
5.1.1	Retentive operands	5-15
5.2	Multi-tasking	5-17
5.2.1	Multitasking applications	5-18
5.2.2	Modules for controlling program execution	5-20
5.3	Start/stop signals	5-21
5.4	Run LED	5-23
5.5	Error handling	5-24
5.5.1	Error handling without error program	5-25
5.5.2	Error handling with error program	5-26
5.5.3	Special handling of I/O errors	5-27
5.5.4	Modules for error handling	5-27
5.5.5	Overview of error numbers	5-28
5.6	The Command Interpreter (CI)	5-31
5.6.1	Connection to a dialog device	5-31
5.6.2	Selecting the command interpreter (Login)	5-33
5.6.3	Exiting the command interpreter	5-35
5.6.4	CI command	5-35
5.6.5	Displaying operands and statuses with Display (D)	5-39
5.6.6	Changing operands with Modify (M)	5-43
5.6.7	Commands for program controller	5-46
5.6.8	Commands for forcing inputs and outputs	5-48
5.6.9	Initialising user memory	5-50
5.6.10	Password	5-51
5.6.11	Driver-specific commands	5-52
5.6.12	Linking CI commands	5-53
5.7	Miscellaneous	5-55
5.7.1	Memory for project files and drivers	5-55
5.7.2	Working memory for programs and drivers	5-56

## 5. Fundamental principles of the FST operating system

**Contents of this chapter**      This chapter provides an overview of the functions featured in the FST PLC operating system.

**Further information**      Extra functions of the PLC operating system can be added by downloaded drivers and modules.



Detailed information on modules and drivers appears in Volume 2.

## 5. Fundamental principles of the FST operating system

### 5.1 Operands (operating resources of the controller)

Programs of a PLC consist of program codes that handle the data. These data are provided by the PLC operating system for the programs in the form of operating resources (Operands). These include the inputs and outputs of the controller, as well as internal memory elements such as flags, indexes and a number of special operands. These are the conventional operands of a PLC.

In principle, a distinction is drawn between operands with the size of only one bit and those with 16 bits, referred to as word. In some cases, the operating resources can be used alternatively as a bit value or in summarised format as the value of a word. For the FST software, these are all inputs, all outputs and all flags. All other operands can be used only either as a word or a bit.

If an operating medium is available either as a bit or in combined format as a word, it is an accepted rule that a word contains the bit with the number 0 as the lowest-value bit (far right). Accordingly, the highest-value bit of the word (far left) is the bit with the number 15.

Other operating resources feature special properties anchored in the operating system. The counters can also be used in the form of a counter value as a counter word CW and in the form of a 1-bit counter status C (not running or running). The situation is similar for timers, which have a timer word TW and a timer status T. The FST PLC operating system ensures that these operating resources will always be changed together.



## 5. Fundamental principles of the FST operating system



The individual operands and how they are used in programs is described below. The following abbreviations are used in the syntax description:

w – word number

b – bit number

u – user number (for fieldbus operands only)

Fieldbus operands exist for the HC1X and HC2X controllers.

### Value ranges of the multi-bit operands

All multi-bit operands are 16-bit values.

Value range	Description
0 to 65535	Decimal without sign
-32768 to +32767	Decimal with sign
\$0000 to \$FFFF	Hexadecimal

Tab. 5/1: Value range of the multi-bit operands

Operands



**Note**  
The table below shows the maximum address range supported by FST. Information on any controller-specific deviations can be found in the documentation on the controller in use.

Inputs

256 possible input words (0...255), each with 16 bit (0...15), can be addressed either as word or bit.

Input bit	
Syntax	Iw.b Iu.w.b *)
Maximum address range	I0.0 ... I255.15 I1.0.0 ... I99.255.15 *)
Operations	Request
*) Only with fieldbus systems	

Tab. 5/2: Input bits

Input word	
Syntax	IWw IWu.w *)
Maximum address range	IW0 ... IW255 IW1.0 ... IW99.255 *)
Operations	Compare
*) Only with fieldbus systems	

Tab. 5/3: Input words

## 5. Fundamental principles of the FST operating system

### Outputs

256 possible output words (0...255), each with 16 bit (0...15), can be addressed either as word or bit.

Output bit	
Syntax	Ow.b Ou.w.b *)
Maximum address range	00.0 ... 0255.15 01.0.0 ... 099.255.15 *)
Operations	Request, set, reset, assign, negate
*) Only with fieldbus systems	

Tab. 5/4: Output bits

Output word	
Syntax	OWw OWu.w *)
Maximum address range	OW0 ... OW255 OW1.0 ... OW99.255 *)
Operations	Load, compare
*) Only with fieldbus systems	

Tab. 5/5: Output words

5. Fundamental principles of the FST operating system

Flag 10,000 flag words (0...9999), each with 16 bit (0...15), can be addressed either as words or bits.

Flag bit	
Syntax	Fw.b
Maximum address range	F0.0 ... F9999.15
Operations	Request, set, reset, assign, negate

Tab. 5/6: Flag bits

Flag word	
Syntax	FWw
Maximum address range	FW0 ... FW9999
Operations	Load, compare

Tab. 5/7: Flag words

Register 256 index (0...255), can be addressed only as word.

Index	
Syntax	Rw
Maximum address range	R0 ... R255
Operations	Load, compare

Tab. 5/8: Index

## 5. Fundamental principles of the FST operating system

### Timer

There are 256 timers (0...255) available. All can be programmed in LDR either as pulse (T), delayed switch-on (TON) or delayed switch-off (TOFF) timers. Only pulse timers (T) are available in STL.

Timer status bit	
Syntax	Tn, TONn, TOFFn
Maximum address range	T0 ... T255 TON0 ... TON255 TOFF0 ... TOFF255
Operations	Request, set, reset, assign, negate

Tab. 5/9: Timer status bits

Timer word	
Syntax	TWn
Maximum address range	TW0 ... TW255
Operations	Load, compare

Tab. 5/10: Timer words

Timer preset	
Syntax	TPn
Maximum address range	TP0 ... TP255
Operations	Load, compare

Tab. 5/11: Timer preset

## 5. Fundamental principles of the FST operating system

The timer type is determined by the designation for the timer operand (timer status) (nn denotes the address of the timer):

- Pulse timer= $Tn$
- Delayed switch-on timer= $TONn$
- Delayed switch-off timer= $TOFFn$



Recommendation: Keep a timer to one type in the project. A delayed switch-off timer should not be used at a different location in the project, e.g. as a pulse timer. Further information on the use of timers can be found in Section A.4.

## 5. Fundamental principles of the FST operating system

### Counters

There are 256 counters (0...255) available.

Counter status bit	
Syntax	Cn
Maximum address range	C0 ... C255
Operations	Request, set, reset, assign, negate, increment, decrement

Tab. 5/12: Counter status bit

Counter word	
Syntax	CWn
Maximum address range	CW0 ... CW255
Operations	Load, compare, increment, decrement

Tab. 5/13: Counter words

Counter preset	
Syntax	CPn
Maximum address range	CP0 ... CP255
Operations	Load, compare

Tab. 5/14: Counter preset



Further information on the use of counters can be found in Section A.3.

5. Fundamental principles of the FST operating system

Constants

Constants are designated with V and must lie within the permissible value range.

Constants	
Syntax	Vn
Value range	Decimal without sign: 0 ... 65535 Decimal with sign: -32768 ... +32767 Hexadecimal: \$0000 ... \$FFFF
Operations	Load, compare

Tab. 5/15: Constants

Function units

There are 256 function units (FU) available. Of these, 7 (FU32 to FU38) are used to transfer parameters. The remainder can be used as required.

Function units	
Syntax	FUn
Maximum address range	FU0 ... FU255
Operations	Load, compare

Tab. 5/16: Function units



In the command interpreter, function units are addressed with 0 (e.g. D0255 for display FU255).

As function units FU32 to FU38 are used to transfer parameters to modules, these exist separately for each program. Each program can reach only its own FUs (32 to 38), the syntax is therefore simply FUw.



5. Fundamental principles of the FST operating system

In the command interpreter, the FUs of all programs can be addressed. The program number must be indicated.  
Op.w (p = program number, w = FU number 32 to 38).

Program and program status

The program and program status operands provide information on the status of the corresponding program.

Program (single-bit)	
Syntax	Pn
Maximum address range	P0 ... P63
Operations	Request, set, reset

Tab. 5/17: Programs

Program status (single-bit)	
Syntax	PSn
Maximum address range	PS0 ... PS63
Operations	Request, set, reset, assign, negate

Tab. 5/18: Program status

The program with the number n can be started and/or stopped via the operand Pn.  
After starting with the SET Pn command, Pn=1 and PSn=1.  
After stopping with the RESET Pn command, Pn=0 and PSn=0. The current program Pn can be halted via RESET PSn.

5. Fundamental principles of the FST operating system

Pn	PSn	Program status
0	0	Inactive
0	1	Reserved
1	0	Active but halted
1	1	Active and running

Tab. 5/19: Program status

Error status and  
error word

The error status specifies whether an error is present.  
The error word contains any fault number.

Error status (single-bit)	
Syntax	E
Values	0: No error 1: Error
Operations	Request, reset

Tab. 5/20: Error status

Error word	
Syntax	EW
Values	0: No error > 0: Error number
Operations	Load, compare

Tab. 5/21: Error word

## 5. Fundamental principles of the FST operating system

### Initial execution flag

The initial execution flag is 1 for a program's first run and then automatically resets to 0.

Initial execution flag (single-bit)	
Syntax	FI (for every program)
Values	0: Program cycle > 1 1: first program cycle
Operations	Request

Tab. 5/22: Initial execution flag

There are separate initial execution flags for each program. The same principles apply for addressing as for function units.

### 5.1.1 Retentive operands

Retentive operands retain their values even when the controller is switched off. The retentive status of FST operands is controller-dependent.

CPX-FEC, HCXX	FEC Compact, FEC Standard
<ul style="list-style-type: none"><li>– FW0 to 9999</li><li>– R0 to 255</li><li>– TP0 to 255</li><li>– C0 to 255</li><li>– CP0 to 255</li><li>– CW0 to 255</li><li>– FU0 to 31, FU39 to 255</li><li>– Password</li></ul>	<ul style="list-style-type: none"><li>– FW0 to 255</li><li>– R0 to 127</li><li>– TP0 to 127</li><li>– C0 to 127</li><li>– CP0 to 127</li><li>– CW0 to 127</li><li>– Password</li></ul>

### **Configuration of the retentive driver on HC1X**

With central units HCOX and HC20 and also FEC Compact, FEC Standard and CPX-FEC, the aforementioned operands are always retained. No driver or other element need be configured. The retentive storage of operands cannot be disabled.



You can use function module F9 (see Volume 2) to reset all operands if required.

If in a project, the operands listed above are to be kept retentive in the HC1X central units, the retentive driver “DRAD” must be entered and parameterised in the project’s driver configuration. Under “Special parameters”, the prefix (“-q”) is usually to be retained. This option prevents a multiple line output onto the screen of the IPC when the driver is started.

5.2 Multi-tasking

The FST runtime system is capable of multi-tasking. It can complete the processing cycles (task) of several programs in succession. While the system is processing the task of one program, the other active programs are not processed. The processing of program parts and moving onto the next program (task change) happens so quickly, however, the programs appear to run in parallel. We call this quasi-parallel program processing. A task change takes place in the following situations:

STL program		LDR program
Step program	Parallel logic program	
– after a step has been processed	– at the end of a program	– at the end of the program and after a jump has been executed
– after a program module CMP has been selected		

Tab. 5/23: Task change

Example

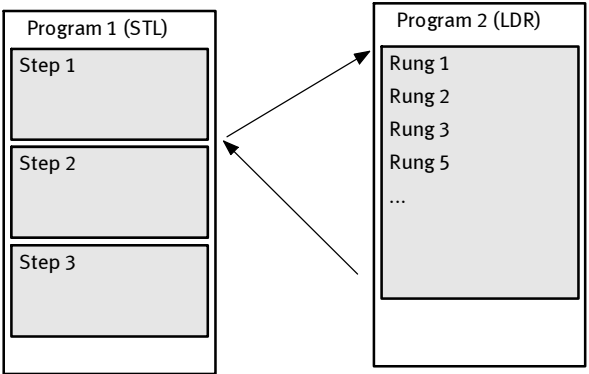


Fig. 5/1: Task change (example)

## 5. Fundamental principles of the FST operating system

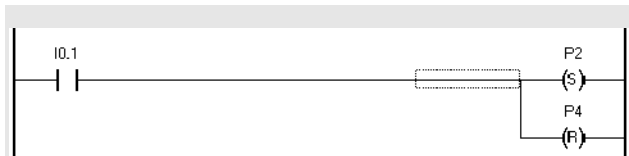
### 5.2.1 Multitasking applications

The program n can be activated (started) and deactivated (stopped) from another program or module. Activation is triggered by setting, deactivation by resetting the Pn operand.

#### Example in STL

```
STEP Call
IF      I0.1
THEN   SET    P2    "Activate program 2
      RESET   P4    "Deactivate program 4
```

#### Example in LDR



A rising edge to I0.1 activates program 2 and deactivates program 4. The program shown here and program 2 run in quasi-parallel.

The LDR symbols --( )-- (assignment) and --( / )-- (negation) cannot be used to activate and deactivate a program. They are not permitted for the P operand.

The quasi-parallel processing of several programs may significantly slow up the execution of the individual programs compared to if they were run alone.



## 5. Fundamental principles of the FST operating system

### Nested program selection

If several programs are running quasi-parallel, the processing sequence always depends on the program number. If one program starts another, the one just started will be activated only (accepted into the list of programs to be processed). It will not be run until it appears at the top of the list.

#### Example 1

Program 0 (STL) starts Program 4 (LDR) in Step 1. Program 4 then starts Program 3 (LDR). The processing sequence is then as follows:

Program 0  
Program 4  
Program 0  
Program 3  
Program 4  
Program 0 ... etc.

In the STL program, only one step is executed at a time, while the LDR programs process until the end (unless they contain jumps or program module calls).

#### Example 2

Program 0 (STL) starts Program 4 (STL) in Step 1. This in turn starts Program 2 in Step 2 (STL). The processing sequence is then as follows:

Program 0	Step 1
Program 4	Step 1
Program 0	Step 2
Program 4	Step 2
Program 0	Step 3
Program 2	Step 1
Program 4	Step 3
Program 0 ... etc.	

## 5. Fundamental principles of the FST operating system

Function and program modules represent a fixed component of the selecting program (as sub-routines). They do not run in parallel to the selecting program. They are processed in the task of the selecting program. After a program module has been selected, a task change always takes place, although not after a function module has been selected (new in FST Version 4!).

The number and type of the programs running in parallel is not restricted.

For diagnostic purposes, the status of a program can be determined in online operation using the display command. The CI responds to the DPn command as follows:

= <Type>,0,<Status>,<Step number>

<Type>: STL=0; LDR, FUP=1; C=2

DPn=	Description of the third parameter
...,0,..	The program is inactive. It is not involved in the task change (Pn=0).
...,2,..	The program is active but halted. It is involved in the task change (Pn=1, PSn=0).
...,3,..	The program is active and running. Its task is being processed (Pn=1, PSn=1).

Tab. 5/24: Description of the <Status> parameter

### 5.2.2 Modules for controlling program execution



Standard modules are provided for controlling program execution (see Volume 2).



5.3 Start/stop signals

With FST, you can use the PLC settings (see Section 2.5) to influence the runtime behaviour of your control. You can, for example:

- specify a stop program (started when the system stops),
- specify whether programs are to be reset or halted when stopping using the start/stop switch,
- configure a start/stop input.



Further information on the new functions can also be found in section Tab. 2/15.

You should configure a start/stop input if, instead of the start/stop switch, you want to use an input for start and stop. The start/stop input and the start/stop switch have the same action.



**Note**  
The start/stop input is active **only** if the start/stop switch of the controller (where present) is set to “Start”.

Phase	Action
<div>– Power on</div> <div>– After bootup</div>	Level-sensitive
<div>– In operation</div>	Edge-sensitive (positive edge for start signal, negative edge for stop signal)

Tab. 5/25: Action

5. Fundamental principles of the FST operating system

With a negative edge (switch from start to stop), either all programs are deactivated (stopped) or halted (interrupted) depending on the “Reset programs” option in the PLC settings (see Tab. 5/26).

With a positive edge (switch from stop to start), Program 0 and any other halted (interrupted) programs are started, regardless of the “Reset programs” option in the PLC settings (see Tab. 5/26).

Start/stop signal		PLC settings “Reset programs”	
		Active (default)	Inactive
Stop	– Online control panel – CI command S (stop)	Deactivate program	
	– Start/stop switch – Start/stop input	Deactivate program	Halt program
Start <sup>1)</sup>	– Online control panel – CI command R (RUN)	– Continue halted programs – Start P0 if active	– Continue halted programs – Start P0 if active
	– Start/stop switch – Start/stop input	– Continue and/or start P0 if active <sup>2)</sup>	– Continue halted programs – Start P0 if active
<sup>1)</sup> Start signal has no influence on active programs that have not been halted. <sup>2)</sup> The program status of other programs halted by online control panel or CI command (BREAK) is not influenced.			

Tab. 5/26: Influence of PLC setting “Reset programs”

Controllers without start/stop switch

If the controller does not feature a start/stop switch and no start/stop input has been configured in the PLC settings (see Tab. 2/15), Program 0 is automatically started after bootup.

### 5.4 Run LED

The Run LED of FEC and HCOX controllers displays the current status of the program processing.



For Run LED of the CPX-FEC, see description for CPX-FEC type P.BE-CPX-FEC-...

Run LED	Description
Green	Programs running
Yellow	No programs are running
Red	A FST error has occurred

Tab. 5/27: Run LED

### 5.5 Error handling

Errors can occur in an automated system. We distinguish between program and system errors. A program error can be, for example, a division by zero. A system error can be, for example, the error in a I/O card.

The FST PLC operation system detects numerous errors and handles them in a pre-programmed manner. You have the option of manipulating error handling and where necessary, also error rectification. An error program is available for these tasks.

If an error occurs during runtime, a value is entered into the error word. This value identifies the error that has occurred (error number). At the same time, two other values are saved that describe the error location. In general, these are the program and step number. If the program contains no steps, the step number is zero. If the error has not occurred in a program, the program number is 255. With I/O errors, the second value shows the base address of the affected card, otherwise zero.

A new error is entered into the error word if this has a value of zero, i.e. no error has previously occurred. This means that only the first error is saved, later ones are ignored.



The CPX terminal has a diagnostic memory that records up to 40 entries (see CPX system description).

If an error output is configured, its condition follows the error word. If the value of the error word is zero, the output is reset, otherwise set. The error output can be requested from the programs in precisely the same way as the remaining outputs.

### 5.5.1 Error handling without error program



With the CPX terminal, you can set the required signal status in the event of a fault by parameterising (Fail safe parameterising).

With FEC Standard, FEC Compact and PS1, the following applies: When you have activated the “Reset outputs” option in the PLC settings, if an error occurs, all programs are stopped and all outputs are reset (apart from the error output).

The error word and additional data can be requested as operand E or EW or with function module F22 (FU32=0). The request does not delete the error. It can be deleted by resetting the operands E and/or EW or with function module F22 (FU32=1).

If E or EW is unequal to zero, it corresponds to the new occurring error (only if EW=0 beforehand).

The error word and error output are deleted by deleting the active project in the controller (Y command) or downloading a project. Starting or stopping the project or individual programs (via CI or start/stop switch) does not delete the error.



Before the project is restarted after an error event, the error should be deleted to enable a new error to be detected.

### 5.5.2 Error handling with error program

The error program is a standard FST program. It cannot, however, be P0. The program that is used as the error program is specified in the PLC settings for the project or by the function module F21. F21 also enables the error program to be converted or deactivated at any time.



Standard modules for error handling, see Volume 2.

If the error program has been defined in the PLC settings, the linker checks whether the program exists. Function module F21 does not perform this check. If a non-existent program is selected, the system responds as if no error program had been defined.

If an error occurs, the error program is selected immediately and exclusively. Programs that were active when the error occurred “sleep” while the error program is active. If the error occurs in a program, the program is stopped before the error program is selected.

The error program can also be a step program. Instead of the normal task change, a simple I/O request is made and no other programs are run. The same applies for parallel logic programs. The program statuses can be changed in the error program (or via CI), although the programs continue to sleep while the error program is being run.

If function module F22 is selected with FU32=2, the “sleeping” programs are “woken”. The task change between the programs is carried out again as usual. All active programs are now run again and continue from the point of interruption. Status changes made by the error program are considered. Please ensure that if the error occurs in a program, the program is stopped before the error program is selected.

## 5. Fundamental principles of the FST operating system

Selecting function module F22 with FU32=2 also deletes the error. If this selection is made before an error program is run, only the error is deleted. However, the error is never deleted automatically.

Selecting F22 with FU32=3 resets all programs and outputs, even if the “Reset programs” option has not been selected in the PLC settings. The error and error output remain unchanged. Regardless of whether the selection was made in the error program or elsewhere and whether an error output has been indicated or an error program defined and/or is active.

If an error occurs while the error program is active, all programs are stopped and all outputs are reset. It is irrelevant whether the error program was started by an error or by normal selection. If no further errors are detected, the new error is entered. If the old error was not deleted, error number 39 “double error” is entered. The two values for additional information contain, as usual, the error location (i.e. the number of the error program and the number of the current step when the error occurs).

### 5.5.3 Special handling of I/O errors

I/O errors 11 and 12 are handled somewhat differently. If error 11 or 12 occurs, no further error 11 or 12 can occur. To re-enable error 11 or 12, function module F25 must be run (see below). I/O errors are also re-enabled if a project is started with the R command.

### 5.5.4 Modules for error handling



Standard modules are provided for error handling (see Volume 2).

### 5.5.5 Overview of error numbers

The FST controller errors are numbered and usually appear with detailed additional information, although sometimes without an explanation. The error messages of the FST PLC operating system are listed below.

The error status can be indicated with CI command “DF”. The response consists of three numbers. The first is the error number, the other two provide additional information on the error situation. The second number is usually the program number and the third the step number.



For further information, see Section 5.6.5 under CI command “DF”.

If the program number is 255, the error did not occur in a program and was caused externally. A typical cause of this error is a faulty I/O card. In this case, the third figure is the number of the input or output word in which the error has occurred. The card can then be easily identified via the I/O configuration. In the case of driver errors, the third figure is the driver number.

For test purposes, an error status can be generated with CI command “MF”. This enables the error handling of a project to be tested.



## 5. Fundamental principles of the FST operating system

Error number	Operating status	Error handling
0	No error	–
2	Checksum error in project file PROJECT.RUN	Reload complete project.
6	Program 0 should have been started but is not present.	Create and download Program 0.
7	Attempt to set or reset an non-existing program or its status.	Modify program or download missing program.
9	Program cannot be started due to an error in the project file PROJECT.RUN.	Check project and reload. A driver should be used although not loaded or a driver is loaded but cannot be used because the relevant conditions are not satisfied. The required driver hardware may not be present or is incorrectly configured. Possibly insufficient working memory in the controller.
11	I/O card faulty, short circuit at output or no power supply.	Replace I/O card, clear short circuit no connect power supply.
12	I/O card not found	Check I/O card; check switch position on the card and in the I/O configuration.
13	Watchdog expired. A driver, module or I/O script has blocked the runtime system for longer than 1 second and triggered a restart.	
14	Driver to be started not found. A required driver cannot be found or cannot run due to an initialisation error. The correct environment (hardware, parameters) may not exist for the driver.	Integrate driver, set parameters correctly in driver configuration and check hardware.
36	Nested CMP/CFM or CMP/CFM not found at selection.	Correct program structure.
39	Double-error, error in program.	Clear error source.
42	CPX diagnosis	Clear CPX error

5. Fundamental principles of the FST operating system

Error number	Operating status	Error handling
57	Project file (PROJECT.RUN) cannot be read.	Reload project.
59	Arithmetical error	Correct program.

Tab. 5/28: Error description



Drivers and modules may cause errors that are not described here. Information can be found in the documentation for the corresponding driver (see Volume). Drivers usually generate error numbers that are calculated as follows:  
Driver number \* 100 + x

## 5.6 The Command Interpreter (CI)

The Command Interpreter, “CI” for short, enables the controllers to be operated externally by a Terminal or Terminal Emulator and represents the interface for online operation of FST.



### Note

The FST contains a Terminal Emulator (see Section 2.9.6) that enables you to send manual CI commands to the Command Interpreter. The term “Command” is used below to mean both command and instruction.

### 5.6.1 Connection to a dialog device

To operate the command interpreter, it must be connected to a suitable dialog device. The following options are available:

- PC with RS232 or TCP/IP port and Terminal Emulator (e.g. CI Terminal of FST, see Section 2.9.6)
- Terminal with RS232 or TCP/IP port.

#### Serial port

The serial port on the controller to be used by the CI for communication can usually be set under “Controller Settings” in the project.



### Caution

With CPU Module HC16: The integrated COM port does not feature a FIFO. With high CPU utilisation, CI commands may be lost.

With high CPU capacity utilisation, use the CI only in combination with an external serial port (COM2 or COM3).

5. Fundamental principles of the FST operating system

Controller	COM	Explanation
CPX-FEC, FEC Standard, FEC Compact	0	Integrated COM port
HCOX	0	Integrated COM port (default)
	1	COM1 on a CP3x module
	2	COM2 on a CP3x module
HC1X	1	Integrated COM1 port (default) or COM1 on a CP3x module, if the integrated COM1 port has been deactivated.
	2	COM2 on a CP3x module
HC2X	1	Integrated COM port (default) or COM1 on a CP3x module, if the integrated COM port has been deactivated
	2	Integrated EXT port (default) or COM2 on a CP3x module, if the integrated EXT port has been deactivated

Tab. 5/29: Numbering of the COM ports

With central units HC1X and HC2X, CI commands – in parallel with communication via serial port – can be entered via the connected keyboard. The response is displayed on the connected screen.

TCP/IP

If the correct FST drivers are installed, the CI can also be accessed via additional COM ports or TCP/IP.



**Note**

Please note that some functions of the additional CI ports are limited.

## 5. Fundamental principles of the FST operating system

### 5.6.2 Selecting the command interpreter (Login)



#### Caution

The command interpreter (CI) contains commands that reorganise or delete parts of the memory. This destroys existing data.

- Only use CI commands if you know their effects!

#### Login with FST

All online functions of the FST use the CI. You can also send manual CI commands via the CI terminal integrated into FST (see Section 2.9.6).

#### Login with Terminal or Terminal Emulator

The CI is registered on a connected terminal after either DC4 (Control T) has been entered or a hardware break has been transferred. Any command currently being processed is cancelled.

```
DC4 (Ctrl T)
```

The controller responds by displaying the version number of the runtime main program and its standard prompt “>” in the next line.

```
FESTO IPC V2.nn  
>
```

5. Fundamental principles of the FST operating system

Transfer of BREAK is accompanied by setting of transfer speed on the controller to 9600 and/or 2400 baud. There are 4 distinct methods:

Method	Description
1	When BREAK is received, the speed is changed in cycles twice to 9600 baud and once to 2400 baud. This is the default method.
2	9600 baud is always set (old method).
3	2400 baud is always set (advisable with slow modem connections and when using Field PC Net alias MpRAM).
4	When BREAK is received, the speed is changed in cycles three times to 2400 baud and twice to 2400 baud.

Tab. 5/30: Methods for speed transfer

All methods enable the baudrate to be set to any speed after login (see CI command MV). After booting, Method 1 is set. Methods 1, 2 and 4 also enable use of a previous FST host software. However, login may occasionally fail. If this is the case, try again. Follow the instructions for FST.

FST knows these new login methods and will try to modify them. The login methods can also be set with function module COM1METH (see Volume 2).

5.6.3 Exiting the command interpreter

The X command frees the serial interface used by CI. This command will only function if it has been entered via the serial interface.

X<CR>

No message is sent by the Command Interpreter.

5.6.4 CI command

The sections below describe the valid CI commands. Drivers can receive their own CI commands (see also Section 5.6.11). These are indicated in Volume 2.

- Command structure
- Each CI command has a defined input format. These include:
- a command letter
  - a parameter (letter or number, depending on parameter)
  - a value (not always required).

Input format	Message from CI
<Command letter>[<Parameter>][=Value]	Dependent on command

Tab. 5/31: Input format and message from CI

Both upper and lower case characters can be entered.  
Conclude entries with <CR>.

Incorrect entries can be changed using the backspace key (Ctrl H) before they are concluded with Enter.

5. Fundamental principles of the FST operating system

Command letter

The table below shows valid command letters:

Command letter	Brief description	
B	BREAK	Program run interrupted
DC4 (Ctrl T)	LOGIN	Login
D	DISPLAY	Displays operands
LC	PASSWORD	Enter/change password
LX	PASSWORD	Password protection on/off
M	MODIFY	Modify: Changes operands
R	RUN	Starts/continues program
S	STOP	Stops program
X	LOGOUT	Enables serial port
Y	INIT	Deletes user memory

Tab. 5/32: Command letters

Parameters

The table below shows the possible parameters.



## 5. Fundamental principles of the FST operating system

Parameters	Meaning	Abbreviation
A[ <YN> . ] <WN> . <BN>	Output bit	<p>Instead of the abbreviation, enter the valid value. The value range is dependent on operand type.</p> <p>           &lt;BN&gt;: Bit number            &lt;BN&gt;: Module number            &lt;DN&gt;: Driver number            &lt;PN&gt;: Program number            &lt;RN&gt;: Register number            &lt;TN&gt;: Timer number            &lt;WN&gt;: Word number            &lt;CN&gt;: Counter number            &lt;YN&gt;: Station number         </p>
AW[ <YN> . ] <WN>	Output word	
B<BN>	Program module	
BF<BN>	Function module	
D	Display format	
E[ <YN> . ] <WN> . <BN>	Input bit	
EW[ <YN> . ] <WN>	Input word	
F	Error word	
M<WN> . <BN>	Flag bit	
MW<WN>	Flag word	
O<WN>	Global function units FU0 ... FU31 and FU39 ... FU255	
O<PN> . <WN>	Local function units FU32... FU38	
P<PN>	Program status	
R<RN>	Index	
S<PN>	Program initialisation flag	
T<TN>	Pulse timer	
TA<TN>	Switch-off delay timer	
TE<TN>	Switch-on delay timer	
TV<TN>	Timer pre-setting	
TW<TN>	Timer word	
V	Baud rate	
Z<CN>	Counters	
ZV<CN>	Counter pre-setting	
ZW<CN>	Counter word	

Tab. 5/33: Parameters

5. Fundamental principles of the FST operating system

Value                                      The permitted values depend on the respective parameter and/or operands.

CI response                              The table below shows the response from the CI to valid and invalid CI commands:

CI command CI response	
Command	⟨Command⟩ “\r”
Response to valid commands	⟨Command⟩ ⟨Response⟩ “\r\n⟩\21”
Response to invalid commands	⟨Command⟩ “\b\r\nACCESS ERROR\r\n⟩\21”

For invalid commands, either “ACCESS ERROR” or (rarely) its abbreviation “ERR” appears. A beep then also sounds in the speaker.

### 5.6.5 Displaying operands and statuses with Display (D)

#### D

#### Display

Display enables you to show the statuses and contents of the operands and also the current status of the programs.

Example: Display Output O0.1

Input

```
>DA0.1
```

Output (example)

```
>DA0.1=0  
>
```

The response from the command interpreter is always displayed in the input line. The entered characters “D”, “A0.1” and “CR” (Enter) are sent to the controller immediately. The controller returns “D”, “A0.1” and the response “=0”. The response is concluded with “CR”, “LF” and “>”.

#### Display commands

DA[ <YN> . ]<WN> . <BN>	Displays output bit
DAW[ <YN> . ]<WN>	Displays output word

DB<BN>	Displays program module
--------	-------------------------

Response: “=<Type>,0,<Status>,<Step>”.

- The first value is the module type, STL=0, LDR/FUP=1 or C=2.
- The second value, memory area, is always 0.

## 5. Fundamental principles of the FST operating system

- The third value indicates the status of the selecting program.
- The final value is the current step number within the module.

DBF<BN>	Displays function module
---------	--------------------------

Response as for DB<BN>.

DD	Shows display format of multi-bit operands
----	--

Response:

“=D” for displaying decimal without sign

“=S” for displaying decimal with sign

“=H” for displaying hexadecimal

DE [ <YN> . ] <WN> . <BN>	Displays input bit
DEW [ <YN> . ] <WN>	Displays input word

DF	Displays error word
----	---------------------

Command DE requests the error status from the controller. If no error has occurred, the controller responds with “=0,0,0”.

Error type	Set-up of the CI response
General errors	=<Error number>,<Program number>,<Step number> <sup>1)</sup>
CPX error (42)	=<42>,<CPX error number>,<CPX module number>
I/O error (11, 12)	=<Error no.>,<255>,<No. of input or output word>
<sup>1)</sup> The error number corresponds to the value of the error word (see also Section 5.5.5); program number in which the error occurred; if the program has no steps (e.g. with LDR programs), Step 0 is displayed.	

For example: “=42,5,1”.

## 5. Fundamental principles of the FST operating system

DM<WN> . <BN>	Displays flag bit
DMW<WN>	Displays flag word

DO<WN>	Displays function unit
--------	------------------------

Global function units FU0 to FU31 and FU39 to FU255 can be displayed.

DO<PN> . <WN>	Displays local function unit
---------------	------------------------------

Local function units FU32 to FU38 can be displayed. There are separate function units for each program.

DP<PN>	Displays program status
--------	-------------------------

The response contains 6 values:

The first value denotes the program type:  
STL=0, LDR/FUP=1 or C=2.

The second value, memory area, is always 0.

The third value denotes the program status:  
0 for inactive, 2 for active but halted, or 3 for active.

The fourth value denotes the step number: unequal to zero for STL step programs and LDR programs with jumps as long as the program is active. If a step program is not active, it is in Step 0.

The final two values denote the numbers and step numbers of the selected module.

DR<RN>	Displays register
--------	-------------------

DS<PN>	Displays program initialisation flag
--------	--------------------------------------

## 5. Fundamental principles of the FST operating system

DT<TN>	Displays status for pulse timer
DTA<TN>	Displays status for switch-off delayed timer
DTE<TN>	Displays status for switch-on delayed timer
DTV<TN>	Displays timer pre-setting
DTW<TN>	Displays timer word

DV	Displays baud rate
----	--------------------

The DV command indicates the current baudrate. Possible values are “=1200”, “=2400”, “=4800”, “=9600”, “=19200”, “=38400” or “=56000”.

DZ<CN>	Displays counter status
DZV<CN>	Displays counter pre-setting
DZW<CN>	Displays counter word

### 5.6.6 Changing operands with Modify (M)

#### M

#### Modify

Modify enables you to change the contents and/or statuses of operands.

- To modify an operand directly, without previous display, enter the required value after the prompt and confirm the entry with Enter <CR>.

Input

```
>MAW1=255
```

Output

```
>MAW1=255
```

- Communicating via RS232 enables you to display the contents and/or the status of the operand beforehand. Enter only command letter M and the operand and then press Enter <CR>.

Input

```
>MAW1
```

Output (example)

```
>MAW1=255:
```

The CI reports the current value. After the colon, you can enter the new value and confirm by pressing <CR>.



The values can be entered in decimal, hexadecimal and signed decimal notation (see Display format).

## 5. Fundamental principles of the FST operating system

### Modify commands

MA[ <YN> . ]<WN>.<BN>={ 0   1 }	Modifies output bit
MAW[ <YN> . ]<WN>=<Value>	Modifies output word

MD={ D   S   H }	Modifies display format
------------------	-------------------------

The display format can be set to decimal without sign “=D”, decimal with sign “=S” or hexadecimal “=H”.

ME[ <YN> . ]<WN>.<BN>={ 0   1 }	Modifies input bit
MEW[ <YN> . ]<WN>=<Value>	Modifies input word

MF=<Value>	Modifies error word
------------	---------------------

The value 0 deletes the current error. Every other value generates the corresponding runtime error.

MM<WN>.<BN>={ 0   1 }	Modifies flag bit
MFW<WN>=<Value>	Modifies flag word

MO<WN>=<Value>	Modifies global function unit
----------------	-------------------------------

Modifies global function units FU0 to FU31 and FU39 to FU255.

MO<PN>.<WN>=<Value>	Modifies local function unit
---------------------	------------------------------

Modifies local function units FU32 to FU38. There are separate function units for each program.



## 5. Fundamental principles of the FST operating system

MR<RN>=<Value>	Modifies register
MT<TN>={ 0   1 }	Modifies pulse timer (start/stop)
MTA<TN>={ 0   1 }	Modifies switch-off delayed timer (start/stop)
MTE<TN>={ 0   1 }	Modifies switch-on delayed timer (start/stop)
MTV<TN>=<Value>	Modifies timer pre-setting
MTW<TN>=<Value>	Modifies timer word

MV=<Baudrate>	Sets baudrate
---------------	---------------

The baudrate can be set using commands “MV=1200”, “MV=2400”, “MV=4800”, “MV=9600”, “MV=19200”, “MV=38400” or “MV=56000”. The value can be shortened to 2 characters, for example “MV=96”.



### Note

“MV=56000” is not valid on FEC and HCOX controllers.

MZ<CN>={ 0   1 }	Sets counter
MZV<CN>=<Value>	Sets counter pre-setting
MZW<CN>=<Value>	Sets counter word





5.6.8 Commands for forcing inputs and outputs



**Note**

The force input/output command is not available on all controller types. The following controller types support this feature:

- CPX-FEC, HC20 and HC1X;
- FEC Standard, FEC Compact and HC0X, from operating system version 2.25 upwards.

All digital inputs and outputs can be forced selectively to 0 or 1. If an input bit is forced to 0 or 1, it can be detected by the programs and the CI. If an output bit is forced to 0 or 1, it cannot be detected by the program and the CI. Further information can be found in Section 2.9.4 under “Forcing Inputs and Outputs”.



The Force table is not retentive. It is automatically deleted by the Y command or by downloaded a project.

The following CI commands are available for forcing I/Os.

YF	Deletes Force table
DAF<WN> . <BN>	Displays output bit

Result:

=0: Forced to 0

=1: Forced to 1

=N: Not forced

## 5. Fundamental principles of the FST operating system

DAWF<WN>	Displays output word
----------	----------------------

Result: “=xxxxxxxxxxxxxxxx”, bitwise with:

=0: Forced to 0

=1: Forced to 1

=N: Not forced

DEF<WN>.<BN>	Displays input bit
--------------	--------------------

Result:

=0: Forced to 0

=1: Forced to 1

=N: Not forced

DEWF<WN>	Displays input word
----------	---------------------

Result: “=xxxxxxxxxxxxxxxx”, bitwise with:

=0: Forced to 0

=1: Forced to 1

=N: Not forced

MAF<WN>.<BN>={ 0   1   N }	Enters output bit into Force table
----------------------------	------------------------------------

=0: Forces to 0

=1: Forces to 1

=N: Do not force

MAWF<WN>={ Value   N }	Enters output word into Force table
------------------------	-------------------------------------

=Value: Forces to this value

=N: Do not force

5. Fundamental principles of the FST operating system

MEF<WN>.<BN>={ 0   1   N }	Enters input word into Force table
----------------------------	------------------------------------

- =0: Forces to 0
- =1: Forces to 1
- =N: Do not force

MEWF<WN>={Value   N }	Enters input word into Force table
-----------------------	------------------------------------

- =Value: Forces to this value
- =N: Do not force

5.6.9 Initialising user memory

Y

Initialising



**Caution**  
The Y! command deletes all project data and drivers from the RAM memory.

Y	Deletes all project data and drivers from the RAM memory, with call-back
Y !	Deletes all project data and drivers from the RAM memory, without call-back

### 5.6.10 Password

The following CI commands enable you to enter, change or delete the password online and also activate or deactivate password protection.



A password consists of between 3 and 20 visible ASCII characters. Separators such as commas, spaces, tab, IBM extended characters etc. are not permitted. Further information on password protection can be found in Section 2.5.4.

LC<old>,<new>	Enter/change password <Old>: Old password
LC,<new>	<New>: New password

When a new password is entered, the old password must always be indicated too. LCTEST,FEC changes the password from TEST to FEC, for instance. If no password previously existed, the old password does not need to be entered. The comma, however, must still be entered e.g. LC,FEC.

LX	Password protection on (logout)
LX<Password>	Password protection off (login)

The LX command is also used to login or logout.

For example, if the password is “FEC”, then:

- password protection is de-activated with LXFEC (login).
- command LX or LX with incorrect password activates password protection (logout).

## 5. Fundamental principles of the FST operating system

### 5.6.11 Driver-specific commands

The FST PLC operating system enables the drivers to receive their own commands. Driver-specific CI commands begin with an exclamation mark “!” and the driver number with <DN>, the command itself then follows.

```
!<DN><Command>
```



Driver-specific CI commands are indicated in Volume 2.

A driver does not necessarily have to have its own commands. Many drivers respond to an empty command with status information. Driver-specific commands generally have a similar set-up to the standard CI commands.

For example, the string driver with the number 3 manages display commands for character chains, in which the corresponding string number is used.

Example:

```
>!3D12='Festo'  
>
```



## 5. Fundamental principles of the FST operating system

### 5.6.12 Linking CI commands

Almost all commands can be linked. The CI processes the command in sequence and the responses are grouped. The command groups must be separated by a semicolon (see Example 1).

The commands of a command group (e.g. successive Display or Modify commands) can be separated by a comma and the command symbol itself (“D” or “M”) is not repeated (see Example 2).

#### Example 1

The commands for starting Program P0 and for requesting the program status are:

```
>RP0  
>DP0=0,0,3,2,0,0  
>
```

The same command sequence links:

Input

```
>RP0;DP0
```

Output (example)

```
>RP0;DP0=0,0,3,2,0,0  
>
```

#### Example 2

R0, FW16 and I0.3 are to be displayed. As individual commands:

```
>DR0=432  
>DMW16=0  
>DE0.3=1
```

## 5. Fundamental principles of the FST operating system

The same command sequence links:

Input

```
>DR0,MW16,E0.3
```

Output (example)

```
>DR0,MW16,E0.3=432=0=1  
>
```

Multi-line commands cannot be linked, e.g. Modify commands with display of current value. Linking is also impossible with commands that are transferred to a driver.

### Mass display

Commands used to display values can be suffixed with a minus sign. 16 successive values are then shown as a mass display. This display method is also valid for bit operands.

Example

The “DR1” command displays register 1.

```
>DR1=0  
>
```

By contrast, the “DR1” command displays registers 1 to 16.

```
>DR1--=0=0=0=0=0=0=0=0=0=0=0=0=0=0=0=0  
>
```

## 5. Fundamental principles of the FST operating system

### 5.7 Miscellaneous

#### 5.7.1 Memory for project files and drivers

The FST PLC operating system uses DOS drives to save the project files and drivers. The drives to be used can be specified in the PLC settings for the project or in the driver settings in the driver configuration. Different drives are available on the various controllers.

Controller	Drives	Memory for project, drivers and other files
CPX-FEC	B: <sup>1)</sup>	800 KB <sup>3)</sup>
	A: <sup>2)</sup> <sup>3)</sup>	–
FEC Compact	B: <sup>1)</sup>	FC20: approx. 90 KB FC34 H01 or later: ca. 370 KB Other: approx. 120 KB
	A: <sup>2)</sup>	–
FEC Standard	B: <sup>1)</sup>	360 KB
	A: <sup>2)</sup>	–
HCOX	B: <sup>1)</sup>	HC02 H02 or later: ca. 370 KB Other: approx. 120 KB
	A: <sup>2)</sup>	–
<sup>1)</sup> The start-up file STARTUP.BAT automatically generated by the FST is also stored here. <sup>2)</sup> Drive A is reserved for system files. User files cannot be stored. <sup>3)</sup> With CPX-FEC, the following drivers do not occupy memory on Drive B, as they are already saved to Drive A in the factory. TCPIP drivers, Web Server drivers, MODBUS drivers.		

Tab. 5/34: Memory for project files and drivers

5. Fundamental principles of the FST operating system

5.7.2 Working memory for programs and drivers

To execute, the project and the drivers must be downloaded to the working memory. The working memory is approximately the following sizes in the controller:

Controller	Size of working memory
CPX-FEC	450 KB <sup>1)</sup>
FEC Compact	290 KB (FC20 30 KB)
FEC Standard	200 KB
HCOX	290 KB
HC1X	430 KB
<sup>1)</sup> When TCP/IP drivers, Web Server drivers and MODBUS drivers are loaded, there is still 250 KB of memory available. These drivers are preselected as standard for new projects.	

Tab. 5/35: Size of working memory

The Table below shows the sizes required by a command in the machine code:

Command	Size in machine code
STL command	10 bytes
LDR command	20 Bytes <sup>*)</sup>
<sup>*)</sup> Integrated edge detection	

Tab. 5/36: Command size in machine code

# Operations (STL and LDR)

## Appendix A

Contents

**A. Operations (STL and LDR) ..... A-1**

A.1 Single-bit operations ..... A-4

    A.1.1 Set/Reset (SET, RESET) ..... A-4

    A.1.2 Swapping operand and single-bit accumulator (SHIFT, STL only) . A-6

    A.1.3 Assigning single bit value ..... A-7

    A.1.4 NOP (No operation) ..... A-9

    A.1.5 Logical linking of bits (N, AND, OR, EXOR) ..... A-10

A.2 Multi-bit operations ..... A-16

    A.2.1 Loading value to the multi-bit accumulator (LOAD, STL only) .... A-16

    A.2.2 Transferring value from the multi-bit accumulator (TO, STL only) . A-17

    A.2.3 Transferring value direct (LOAD TO, LDR only) ..... A-17

    A.2.4 Arithmetical operations (+, -, \*, /, <, <=, =, >=, >, < >, INC, DEC) .... A-18

    A.2.5 Logical linking of words (AND, OR, EXOR) ..... A-22

    A.2.6 Conversion (SWAP, BID, DEB) ..... A-26

    A.2.7 Bit shift operations (SHL, SHR, ROL, ROR) ..... A-30

    A.2.8 1's and 2's complement (INV, CPL) ..... A-33

A.3 Standard counter (C...) ..... A-37

A.4 Timer (T...) ..... A-44

A.5 Module call (CMP, CFM) ..... A-52

A.6 Jump (JMP TO) ..... A-54

## A. Operations (STL and LDR)

Contents of this chapter      This chapter provides an overview of the possible STL and LDR operations.

Further information      Basic information on each programming language and a brief overview of the STL and LDR operations can be found in Chapter 3 (STL) and 4 (LDR).



The drivers and modules supplied with the FST runtime library provide a host of other operations. Detailed information on this can be found in Volume 2.

## A.1 Single-bit operations

### A.1.1 Set/Reset (SET, RESET)

#### SET

Setting operand logically to 1 (saving)

STL

In STL, the SET instruction is used.

#### Example

```
IF          I1.0      "If I0.1 = 1
THEN SET    O1.0      "then set O0.1
```

If Input I0.1 delivers 1 signal, Output O0.1 is set with saving.

LDR

In LDR, the coil is identified with S.

#### Example



With a positive edge at Input I0.1, the output O0.1 is set with saving. Otherwise, the status of the output is not influenced.



## RESET

Resetting operand logically to 0 (saving)

STL

In STL, the RESET instruction is used.

### Example

```
IF          I1.0      "If I0.1 = 1
THEN RESET  O1.0      "then reset O0.1
```

If Input I0.1 delivers 1 signal, Output O0.1 is reset with saving.

LDR

In LDR, the coil is identified with R.

### Example



With a positive edge at Input I0.1, the output O0.1 is reset with saving. Otherwise, the status of the output is not influenced.

### A.1.2 Swapping operand and single-bit accumulator (SHIFT, STL only)

#### SHIFT

#### Shifting single-bit operand

##### STL

The SHIFT instruction swaps the indicated single-bit operand for the value in the single-bit accumulator. This command can be used to create various lengths of shift register, longer or shorter than the 16-bit word.

First the single-bit accumulator (EBA) must be loaded and then any SHIFT instruction can be executed.

#### Example

```
STEP 10
  IF      I1.0      "
  THEN LOAD  I1.1    " load I1.1 to EBA
        SHIFT O1.1  " swap EBA and O1.1
        SHIFT O1.2  " swap EBA and O1.2
        SHIFT O1.3  " swap EBA and O1.3
        SHIFT O1.1  " swap EBA and O1.4
STEP 20
  IF      N      I1.0  " wait until input
                        " is deactivated
  THEN JMP TO 10      " repeat
```

As soon as Input I1.0 delivers a 1 signal, the signal states of Outputs O1.1 to O1.4 are changed.

Output O1.1 accepts the state of Input I1.1.

Output O1.2 accepts the old state of Output O1.1.

Output O1.3 accepts the old state of Output O1.2.

Output O1.4 accepts the old state of Output O1.3.

A.1.3 Assigning single bit value

Assignment

Assignment (not saving)

The state of the input signal is assigned to an output. The output follows each signal change at the input.

STL

In STL, an assignment is generated by combining instructions N, SET and RESET (see also Negation).

Example

STEP 1			
IF	N	I1.0	"If I1.0 = 0
THEN	RESET	O1.0	"Reset O1.0
	JMP TO	1	
OTHRW	SET	O1.0	"Set O0.1
	JMP TO	1	

LDR

In LDR, the assignment is carried out using the corresponding coil symbols. The operand follows each signal change in the condition part.

Example



LOAD

Loading value to the single-bit accumulator (STL)



To load values to the multi-bit accumulator, see Section A.2.1. Source and target operand must be of the same type (either single or multi-bit operand).

STL

In STL, the LOAD instruction can be used to load the value of a single-bit operand to the single-bit accumulator. Any number of single-bit operations can then be applied to the single-bit accumulator. To conclude, the result is transferred to the target operands by the TO instruction.

Example

THEN	LOAD	I0.1	"Load value
	AND	I0.2	"AND link with I0.2
	TO	F0.1	"Assign result

TO

Transfer result from the single-bit accumulator to a single-bit operand (STL only)



For transferring single-bit values, see Section A.2.2. Source and target operand must be of the same type (either single or multi-bit operand).

STL

The TO instruction is used to transfer the content of the single-bit accumulator to the target operand (for example, see Section A.1.3, LOAD operation).

A.1.4 NOP (No operation)

**NOP** No operation

Use this instruction if you want to execute without an input condition.

STL In STL, the NOP instruction is used.

**Example**

IF		NOP
THEN	SET	F1.0

LDR In LDR, NOP is transferred as an operand via the contact.

**Example**



Or no condition symbol is entered.



A.1.5 Logical linking of bits (N, AND, OR, EXOR)

N

Negation

With negation, the state of an input signal is assigned negated to an output.

Input	Output
1	0
0	1

Tab. A/37: “Negation” truth table

STL

In STL, a negation is generated by combining instructions N, SET and RESET.

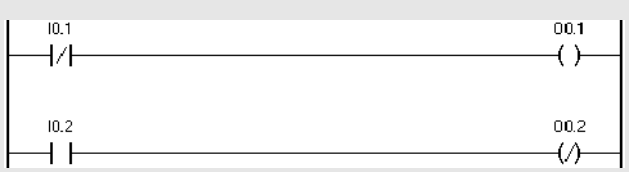
Example

STEP 1			
IF	N	I1.0	“If I0.1 = 0
THEN	SET	O1.0	“Set O0.1
	JMP TO 1		
OTHRW	RESET	O1.0	“Reset O0.1
	JMP TO 1		

LDR

In the LDR, the negation (/) can be carried out at the contact symbol (see rung 1) and at the coil (see rung 2).

Example



AND

AND link

The AND link links together two or more expressions. The link result is true if all AND-linked expressions are true.

Expression 1	Expression 2	Link result
0	0	0
0	1	0
1	0	0
1	1	1

Tab. A/38: Truth table “AND link (AND)”

STL

The AND instruction can AND-link two or more single-bit operands in the condition part.

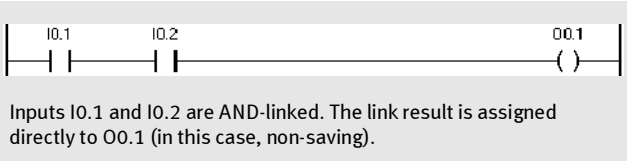
Example

IF		I1.1	"If I0.1 = 1
	AND	I1.2	"And I0.2 = 1
THEN	SET	O2.0	"Set O2.0
OTHRW	RESET	O2.0	"Reset O2.0

LDR

The AND link is shown in the LDR by cascaded locks.

Example



OR

OR link

The OR link links together two or more expressions. The link result is true if one of the OR-linked expressions are true.

Expression 1	Expression 2	Link result
0	0	0
0	1	1
1	0	1
1	1	1

Tab. A/39: Truth table “OR link (OR)”

STL

The OR instruction can OR-link two or more single-bit operands in the condition part.

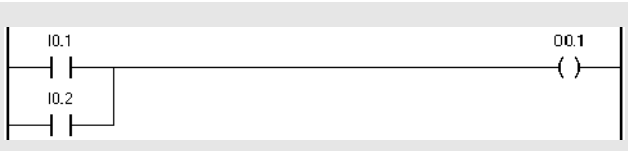
Example

IF		I1.1	“If I0.1 = 1
	OR	I1.2	“OR I0.2 = 1
THEN	SET	O1.5	“Set O0.1

LDR

The OR link is shown in the LDR program by parallel loads. The link result is 1 (true), if at least 1 condition element is true and/or delivers 1 signal.

Example





EXOR

EXCLUSIVE OR link

With the EXCLUSIVE OR link, the link result is 1 (true), if only 1 condition element is true and/or delivers 1 signal.

Input A	Input B	Output
0	0	0
0	1	1
1	0	1
1	1	0

Tab. A/40: Truth table “EXOR link (XOR)”

STL  
An EXOR link can be generated from a combination of AND, OR and NOT.

Example

IF	(	I0.1		"If I0.1 = 1
	AND	N	I0.2 )	"And I0.2=0
	OR	(	N I0.1	"Or I0.1 = 0
	AND	I0.2 )		"And I0.2=1
THEN	SET	O0.1		"Then set O0.1

LDR  
The EXOR link is shown in the LDR program as follows.

Example



### NOT AND link

This link is equivalent to an AND link with negation (/) at the output. The link result is 1 (true), if 1 condition element or all condition elements delivers 0 signal.

Expression 1	Expression 2	Link result
0	0	1
0	1	1
1	0	1
1	1	0

Tab. A/41: Truth table “NOT AND link (NAND)”

### NOT OR link

This link is equivalent to an OR link with negation (/) at the output. The link result is 1 (true), if 1 all condition elements deliver a 0 signal.

Expression 1	Expression 2	Link result
0	0	1
0	1	0
1	0	0
1	1	0

Tab. A/42: Truth table “NOT OR link (NOR)”

### NOT EXCLUSIVE OR link

This link is equivalent to an EXOR link with negation (/) at the output. The link result is 1 (true), if both condition elements deliver the same signal.

Input A	Input B	Output
0	0	1
0	1	0
1	0	0
1	1	1

Tab. A/43: Truth table “NOT EXOR link (XNOR)”

A.2 Multi-bit operations

A.2.1 Loading value to the multi-bit accumulator (LOAD, STL only)

LOAD

Loading value to the multi-bit accumulator



For loading single-bit operands, see Section A.1.3. Source and target operand must be of the same type (either single or multi-bit operand).

STL

In STL, the LOAD instruction can be used to load the value of a multi-bit operand to the multi-bit accumulator. Any number of multi-bit operations can then be applied to the multi-bit accumulator. To conclude, the result is transferred to the target operands by the TO instruction.

Example

IF		I0.1	
THEN	LOAD	V12	"Source operand
	+	FW1	"add FW1
	TO	R0	"Target operand R0

If there is 1 signal at Input I0.1, the value 12 is loaded to the multi-bit accumulator. FW1 is then added. The result is then transferred to Register R0.

## A. Operations (STL and LDR)

### A.2.2 Transferring value from the multi-bit accumulator (TO, STL only)

#### TO

#### Transferring value to a multi-bit operand



For transferring single-bit values, see Section 1.3. Source and target operand must be of the same type (either single or multi-bit operand).

#### STL

The TO instruction is used to transfer the content of the multi-bit accumulator to the target operand (for example, see Section A.2.1, LOAD operation).

### A.2.3 Transferring value direct (LOAD TO, LDR only)

#### LOAD TO

#### Transferring value direct

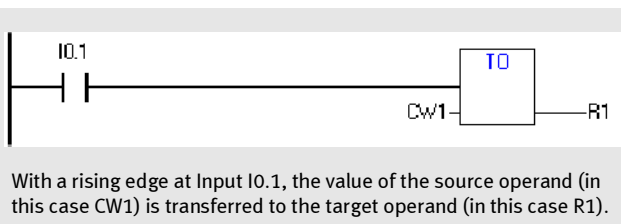


The two instructions LOAD and TO available in STL are grouped into one operation in LDR.

#### LDR

In LDR, the TO instruction can be used to transfer the value of a multi-bit operand to the target operand. Further operations can then be applied to the target operand.

#### Example



### A.2.4 Arithmetical operations (+, -, \*, /, <, <=, =, >=, >, < >, INC, DEC)

**+, -, \*, /, <, <=, =,  
>=, >, < >**

#### Basic calculation methods and comparisons

The four basic calculation methods are applicable to whole numbers preceded with a sign between +32767...-32768.



#### Note

An overrun **cannot** be requested.

#### STL

The four basic calculation methods can be used in the condition part and the executive part.

If the operation is used in the condition part, the result can be compared with a third operand.

If the operation is used in the executive part, the result can be transferred to the target operand with the TO command.

#### Example (basic calculation method in the condition part)

```
IF          (  CW1      "Counter word CW1
      +      CW2  )  "Addition with CW2
      >      V100    "Greater than 100?
THEN...
```

It is confirmed whether the total of CW1 and CW2 is greater than 100.

#### Example (basic calculation method in the executive part)

```
IF...
THEN  LOAD  (  CW1      "Load counter word CW1
      +      CW2  )  "Addition with CW2
      TO      R30    "Transfer to R30
```

CW1 and CW2 are added. The result is transferred to Register 30.

## A. Operations (STL and LDR)

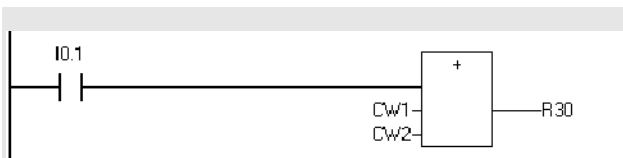
### LDR

With a rising edge of the link result, the value of the multi-bit operand located at the top on the input (left) is loaded to the multi-bit accumulator and then added to the lower multi-bit operand. The result is transferred to the target operand on the output (right).



The AND link of multi-bit operands can only be used in the ladder diagram in the executive part.

#### Example



See “Example STL-Arithmetical Operation in the Executive Part”. However, the operation is carried out only with a rising edge of the link result.

INC, DEC, I, D

INC, I (increment), DEC, D (decrement)

Operation		Description
STL	LDR	
INC	I	Count forwards (increase value by 1)
DEC	D	Count backwards (decrease value by 1)

Tab. A/44: Operation names in STL and LDR

Unlike with other arithmetical operations, these operations are carried out directly without loading the operand to be incremented or decremented to the multi-bit accumulator.



Operations INC, I and DEC, D are usually used in combination with counters. Detailed information on this can be found in Section A.3.

The counter status (C) can be entered as the operand instead of the counter word (CW). The operations are, however, applied to the corresponding counter word.

STL

Operations INC and DEC can be used only in the executive part and are applied directly to the indicated multi-bit operands.

**Example**

```
IF      ...
THEN   INC      R9      "Increase Register 9 by 1
IF      ...
THEN   DEC      R9      "Decrease Register 9 by 1
```

The content of Register 9, independent of the programmed condition, is increased by 1.





Operations INC and DEC can be replicated by operations + and - respectively.

### Example

```
IF      ...
THEN   LOAD  R9      "Load Register 9 to the
                     "multi-bit accumulator
        +    V1      "add 1
        TO    R9      "Transfer result to
                     "Register 9
```

The content of Register 9, independent of the programmed condition, is increased by 1.

## LDR

With a rising edge of the link result, the value of the indicated multi-bit operand is incremented (I) or decremented (D).

### Example



With a rising edge at Input I0.0, the content of Register R9 is increased by 1.

A.2.5 Logical linking of words (AND, OR, EXOR)

AND

AND linking of multi-bit operands

The AND link of multi-bit operands links together the individual bits of two or more multi-bit operands together. The link result is true if all AND-linked bits are true (see also Section A.1.5).



The AND link of multi-bit operands is frequently used for masking or filtering. The bits to be “filtered out” are linked with 1 AND. The other bits are linked with 0 AND and therefore to 0.

Off

High byte								Low byte							
0	0	1	1	1	0	0	1	1	1	0	0	1	0	0	1

and (V15)

High byte								Low byte							
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

becomes

High byte								Low byte							
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1

Fig. A/1: AND link (example)

## A. Operations (STL and LDR)

### STL

The AND link of multi-bit operands can be used in the condition part and in the executive part.

If the operation is used in the condition part, the result can be compared with a third operand. If the operation is used in the executive part, the result can be transferred to the target operand with the TO command.

#### Example (AND in the condition part)

```
IF          ( IW0      "Input word WI0
              AND      V15 ) "mask (00001111)
              >        V0   "Is an input (0.0
                           "to 0.3) active?

THEN      ...
```

It is confirmed whether one of the inputs 0.0 to 0.3 delivers 1 signal. The AND link with constant V15 filters out the contents of the first 4 bits.

#### Example (AND in the executive part)

```
IF      ...
THEN    LOAD  ( R1      "Load Register 1
          AND   R2 )    "AND link with R2
          TO    R10     "Transfer to R10
```

The contents of Register R1 and Register R2 become AND linked. The result is transferred to Register R10.

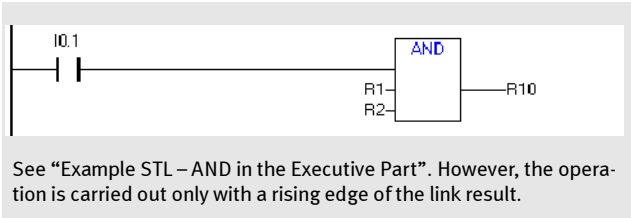
### LDR

With a rising edge of the link result, the value of the multi-bit operand located at the top on the input (left) is loaded to the multi-bit accumulator and then AND linked to the lower multi-bit operand. The result is transferred to the target operand on the output (right).



The AND link of multi-bit operands can only be used in the ladder diagram in the executive part.

Example



OR OR linking of multi-bit operands

As for AND link. Each individual bit, however, is OR linked. The link result is true if one of the OR-linked bits is true.

Off

High byte								Low byte							
0	0	1	1	1	0	0	1	1	1	0	0	1	0	0	1

and

High byte								Low byte							
0	1	1	0	0	1	0	1	0	0	0	0	1	1	1	1

becomes

High byte								Low byte							
0	1	1	1	1	1	0	1	1	1	0	0	1	1	1	1

Fig. A/2: OR link (example)

EXOR

EXOR linking of multi-bit operands

As for OR link. Each individual bit, however, is EXOR linked.  
The link result is true if one of the EXOR-linked bits is true.

Off

High byte								Low byte							
0	0	1	1	1	0	0	1	1	1	0	0	1	0	0	1

and

High byte								Low byte							
0	1	1	0	0	1	0	1	0	0	0	0	1	1	1	1

becomes

High byte								Low byte							
0	1	0	1	1	1	0	0	1	1	0	0	0	1	1	0

Fig. A/3: EXOR link (example)

A.2.6 Conversion (SWAP, BID, DEB)

SWAP

SWAP (swap high and low bytes)

Swaps the contents of the low byte and the high byte in the multi-bit accumulator (see Fig. A/4).



On devices manufactured by other suppliers, the low byte and high byte are sometimes arranged the other way around. The SWAP operation can be used to modify the byte assignment of a value.

Off

High byte								Low byte							
1	1	1	1	1	1	1	1	0	1	0	1	0	1	0	1

becomes

High byte								Low byte							
0	1	0	1	0	1	0	1	1	1	1	1	1	1	1	1

Fig. A/4: Swapping low byte and high byte (example)

STL

The required value is loaded to the multi-bit accumulator beforehand. Following the SWAP instruction, other operations can be carried out with the contents of the multi-bit accumulator. To conclude, the result is transferred to the target operand.

## A. Operations (STL and LDR)

### Example

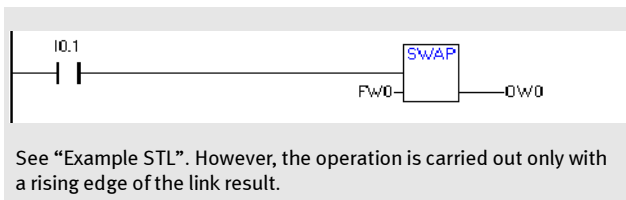
IF		I0.1	
THEN	LOAD	FW0	"Source operand FW0
	SWAP		
	TO	OW0	"Target operand OW0

If there is 1 signal at Input I0.1, the flag word FW0 is loaded to the multi-bit accumulator. Here, the higher value and lower value byte are swapped. The result is transferred to output word OW0.

## LDR

With a rising edge of the link result, the value of the source operand located at the top on the input (left) is loaded to the multi-bit accumulator. Here, the contents of the low byte and the high byte are swapped (see Fig. A/4). The result is transferred to the target operand on the output (right).

### Example



See "Example STL". However, the operation is carried out only with a rising edge of the link result.

**BID**

BID (converging BINARY to BCD display)

Converts the content of the multi-bit accumulator from BINARY into BCD display (binary-coded decimal).



With the decimal display, each of the 4 bits represents a decimal value. Devices often operate in BCD format, these communicate with the PLC/IPC via digital I/Os, e.g. BCD displays and servomotor controller.

Off 2503 <sub>BINARY</sub>

High byte								Low byte							
0	0	0	0	1	0	0	1	1	1	0	0	0	1	1	1

becomes 2503 <sub>BCD</sub>

Figure 4 (2)				Figure 3 (5)				Figure 2 (0)				Figure 1 (3)			
0	0	1	0	0	1	0	1	0	0	0	0	0	0	1	1

Fig. A/5: Converging BINARY to BCD display (example)

STL

The required value is loaded to the multi-bit accumulator beforehand. Following the BID instruction, other operations can be carried out with the contents of the multi-bit accumulator. To conclude, the result is transferred to the target operand.



### Example

```

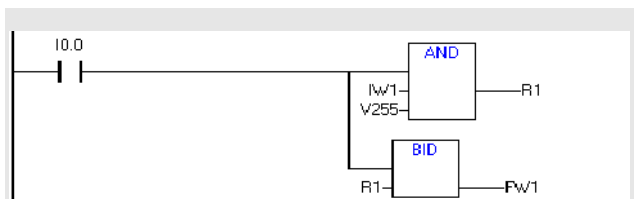
IF          I0.0      "Start key
THEN LOAD   ( IW1     "Source operand IW1
  AND      V255      ) "Mask bit 8...15
  BID                      "convert to BCD
  TO        FW1      "Target operand FW1
  
```

The value in the low byte of input word IW1 is to be converted to BCD display. For this, input word IW1 is first AND-linked with constant V255, in order to mask Inputs I1.8 to I1.15 (bit 8...15 becomes 0). The content of the multi-bit accumulator is converted to BCD display and the result loaded to flag word FW1.

### LDR

With a rising edge of the link result, the value of the source operand located at the top on the input (left) is loaded to the multi-bit accumulator. Here, BINARY is converted to BCD display (see Fig. A/8) The result is transferred to the target operand on the output (right).

### Example



See “Example STL”. However, the operation is carried out only with a rising edge of the link result and Register R1 is used as the clipboard.

DEB

DEB (converts BCD to BINARY display)

As for BID (convert BINARY to BCD display). However, the conversion occurs in the opposite direction from BCD to BINARY display.

A.2.7 Bit shift operations (SHL, SHR, ROL, ROR)

SHL

SHL (Shift to left)

Shifts the content of the multi-bit accumulator by one bit space to the left. The highest-value bit (bit 15) is lost and the right, free bit space (Bit 0) is filled with a zero (see Fig. A/6).



The SHL instruction is equivalent to multiplying by 2.

Off

High byte								Low byte							
0	1	0	1	0	1	0	1	1	1	1	1	1	1	1	1

becomes

High byte								Low byte							
1	0	1	0	1	0	1	1	1	1	1	1	1	1	1	0

Fig. A/6: Shift to left (example)

STL

The required value is loaded to the multi-bit accumulator beforehand. Following the SHL instruction, other operations can be carried out with the contents of the multi-bit accumulator. To conclude, the result is transferred to the target operand.

## A. Operations (STL and LDR)

### Example

IF		I0.1	
THEN	LOAD	FW0	"Source operand FW0
	SHL		
	TO	OW0	"Target operand OW0

If there is 1 signal at Input I0.1, the flag word FW0 is loaded to the multi-bit accumulator. The content of the multi-bit accumulator is then shifted by one bit space to the left. The result is transferred to output word OW0.

### LDR

With a rising edge of the link result, the value of the source operand located at the top on the input (left) is loaded to the multi-bit accumulator. The content of the multi-bit accumulator is then shifted by one bit space to the left. The result is transferred to the target operand on the output (right).

### Example



See "Example STL". However, the operation is carried out only with a rising edge of the link result.

SHR

SHR (Shift to right)

As for SHL (Shift to left). However, the SHR operation shifts the content of the multi-bit accumulator to the right instead of the left. The lowest-value bit (bit 0) is lost and the left, free bit space (bit 15) is filled with a zero (see).



The SHR instruction is equivalent to dividing by 2.

ROL

ROL (Rotate to left)

As for SHL (Shift to left). However, the content of the multi-bit accumulator is rotated by one bit space to the left instead of one bit space to the right. The highest-value bit (bit 15) is transferred to the lowest-value bit position (bit 0).

Off

High byte								Low byte							
1	0	1	0	1	0	1	0	1	1	1	0	1	1	1	0

becomes

High byte								Low byte							
0	1	0	1	0	1	0	1	1	1	0	1	1	1	0	1

Fig. A/7: Rotate to left (example)

ROR

ROR (Rotate to right)

As for ROL (Rotate to left). However, the ROR operation rotates the content of the multi-bit accumulator to the right instead of the left. The lowest-value bit (bit 0) is transferred to the highest-value bit position (bit 15).

A.2.8 1's and 2's complement (INV, CPL)

INV

INV (Inversion – 1's complement)

Inverts each bit in the multi-bit accumulator, therefore forms the 1's complement (see Fig. A/8).



If this function is used with signed integers, it is equivalent to multiplying a number by -1 and then adding -1.

Off

High byte								Low byte							
0	0	1	1	1	0	0	1	1	1	0	0	1	1	1	1

becomes

High byte								Low byte							
1	1	0	0	0	1	1	0	0	0	1	1	1	0	0	0

Fig. A/8: Inverting (example)

STL

The required value is loaded to the multi-bit accumulator beforehand. Following the INV instruction, other operations can be carried out with the contents of the multi-bit accumulator. To conclude, the result is transferred to the target operand.

**Example**

IF		I0.1	
THEN	LOAD	FW0	"Source operand FW0
	INV		
	TO	OW0	"Target operand OW0

If there is 1 signal at Input I0.1, the flag word FW0 is loaded to the multi-bit accumulator. Here, the higher value and lower value byte are swapped. The result is transferred to output word OW0.

LDR

With a rising edge of the link result, the value of the source operand located at the top on the input (left) is loaded to the multi-bit accumulator. Each bit is inverted here (see Fig. A/8). The result is transferred to the target operand on the output (right).

**Example**



See “Example STL”. However, the operation is carried out only with a rising edge of the link result.

CPL

CPL (2’s complement)

Forms the 2’s complement. In the multi-bit accumulator, all bits are inverted (1’s complement) and then added to 1.



In the case of signed integers, this is equivalent to multiplying by -1.

Off

High byte								Low byte							
0	0	1	1	1	0	0	1	1	1	0	0	0	1	1	1

becomes

High byte								Low byte							
1	1	0	0	0	1	1	0	0	0	1	1	1	0	0	1

Fig. A/9: 2’s complement (example)

STL

The required value is loaded to the multi-bit accumulator beforehand. Following the CPL instruction, other operations can be carried out with the contents of the multi-bit accumulator. To conclude, the result is transferred to the target operand.

Example

```
IF      ( R32      "Confirm whether R32
  <     V0      )  "negative
THEN  LOAD      R32      "Source operand R32
      CPL
      TO        R1      "Target operand R1
```

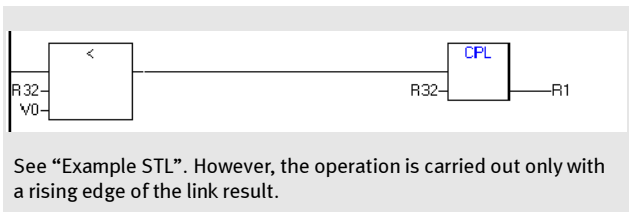
In the example below, the program confirms whether Register 32 contains a negative value. If so, it converts this negative value in a positive one and saves it to Register 1.

## A. Operations (STL and LDR)

### LDR

With a rising edge of the link result, the value of the source operand located at the top on the input (left) is loaded to the multi-bit accumulator. The 2's complement is inverted here (see Fig. A/8). The result is transferred to the target operand on the output (right).

#### Example





### A.3 Standard counter (C...)



Information on the special fast or interrupt-controlled counters on various PLC/IPCs can be found in the hardware manual for the corresponding PLC/IPC and in Volume 2 “Drivers and modules”.

There are 256 counters available. Three operands are assigned to each counter. They provide information on the state, the current counter status and the volume to be counted (preset value) of the counter.

Operands of the standard counters (LDR, STL)		
Operands	Designation	Description
C0 ... C255	Counter status	Single-bit operand that shows whether the counter is activated or deactivated. 1: Counter is activated 0: Counter is deactivated (stopped or expired) – If the counter status (Cnn) is set to <sup>1)</sup> the corresponding counter word automatically resets to 0. This applies also if the counter was already active. – The counter status (Cnn) is automatically reset to 0, if: – the counter word reaches the forward values of the counter preset – the counter word is counted back to 0 on backwards counting.
CP0 ... CP255	Counter preset	Multi-bit operand that contains the counter preset: – with forwards counter, the final value – with backwards counter, the start value
CW0 ... CW255	Counter word	Multi-bit operand that indicates the current status of the counter.
<sup>1)</sup> nn represents the address of the counter (0...255).		

Tab. A/45: Operands of the standard counters

Standard counters can be used as:

- Forwards counter
- Backwards counter

With forwards counting, the standard value is 0 and it is incremented from the current counter status. With backwards counting, the standard value is > 0 and it is decremented.

	Forwards counter	Backwards counter
Start value	0	12345 *)
.	↓	↓
.		
.		
Final value	12345 *)	0
*) Counter preset (example)		

Tab. A/46: Start and final value of a counter

Before a counter is used, this value must be initialised as a forwards or backwards counter.

**Forwards counter**

Using counter as forwards counter

**Initialising counter as forwards counter**

Initialise a counter as a forwards counter as follows:

1. Load final value of the counter into the counter preset (CPnn).
2. Activate counter by setting counter status (Cnn).  
The corresponding counter word (CWnn) is reset to 0.

### Count

The following count functions are available:

- Count forwards (INC, I)
- Decrement (DEC, D)



As operands, you can indicate the counter word (CWnn) or the counter status (Cnn). Further information on operations INC, I and DEC, D can also be found in section A.2.4.

### STL

The forwards counter is initialised by the following operations:

- LOAD... TO loads the final value into the counter preset.
- SET activates the counter.  
The corresponding counter word (CWnn) is reset to 0.

#### Example: Initialising forwards counter

```
IF      AND      N      I0.0
      THEN LOAD      C3
      TO      V100      "Final value 100
      SET      CP3      "Counter preset
      C3      "CW3 becomes 0
... 
```

The counter is initialised by a 1 signal at Input I0.0 when it has been deactivated. In this case, the value 100 is loaded into the counter preset and counter (C3) is activated. The counter word (CW3) is automatically reset to 0.

Operations INC and DEC increment and decrement the counter.

### Example: Count

```
IF      ...  
THEN   INC   CW3      "Increment  
...    ...    "counter word 3  
or  
...  
THEN   DEC   C3       "Increment  
...    ...    "counter word 3
```

Counter word CW3 is incremented by 1 and/or decremented by 1.

### LDR

The forwards counter is initialised by the counter box. When the link result has an rising edge, the counter box carries out the following operations:

- The value of the source operand located at the input (left) is loaded to the counter presetting (CPnn) of the above counter.
- The counter is activated (counter status (Cnn) is set.

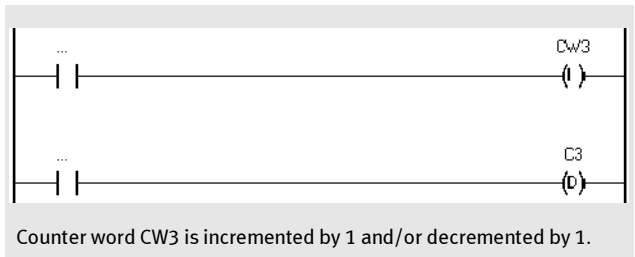
### Example: Initialising forwards counter



The counter is initialised by a positive edge at Input I0.0 when it has been deactivated. The value 100 is loaded to the counter preset (CP3) and counter (C3) is activated. The counter word (CW3) is automatically reset to 0.

Operations INC and DEC increment and decrement the counter.

### Example: Count



## Backwards counter

Using counter as backwards counter

### Initialising backwards counter

A backwards counter is initialised as follows:

1. Activate counter by setting counter status (Cnn). The corresponding counter word (CWnn) is reset to 0.
2. Load final value of the counter into the counter word (CWnn).

### Count

The following count functions are available:

- Count forwards (INC, I)
- Decrement (DEC, D)



As operands, you can indicate the counter word (CWnn) or the counter status (Cnn). Further information on operations INC, I and DEC, D can also be found in section A.2.4.

STL

The backwards counter is initialised by the following operations:

- SET activates the counter.  
The corresponding counter word (CWnn) is reset to 0.
- LOAD... TO loads the start value into the counter word.

**Example: Initialising backwards counter**

```
IF          AND      N      I0.0
THEN        SET      C3
            LOAD      V100
            TO        CW3
            ...
            "Activate counter 3
            "CW3 becomes 0
            "Final value 100 to
            "Load counter word
            "CW3 becomes 100
```

The counter is initialised by a 1 signal at Input I0.0 when it has been deactivated. The counter (C3) is activated and the word 100 is then loaded to the counter word.

Operations INC and DEC increment and decrement the counter.

**Example: Count**

```
IF
THEN  INC  ...
      CW3  "Increment
...    "counter word 3
or
...
THEN  DEC  C3  "Increment
...    "counter word 3
```

Counter word CW3 is incremented by 1 and/or decremented by 1.

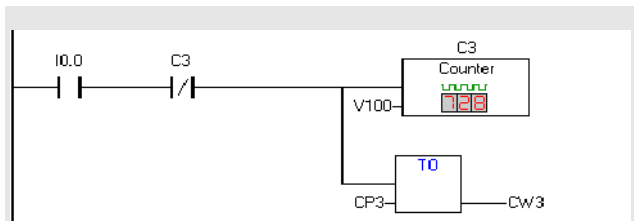
## A. Operations (STL and LDR)

### LDR

To initialise a backwards counter, the operation LOAD TO is used in parallel to the counter box.

- The counter box loads the required counter preset of the above counter and activates the counter (counter status (Cnn) is reset).
- The LOAD TO operation loads the counter preset to the counter word.

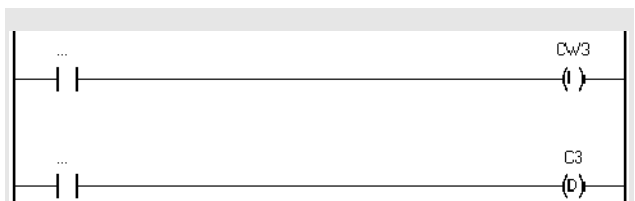
#### Example: Initialising backwards counter



The counter is initialised by a positive edge at Input I0.0 when it has been deactivated. The value 100 is loaded to the counter preset (CP3) and counter (C3) is activated. The counter preset is then loaded to the counter word (CW3).

Operations INC and DEC increment and decrement the counter.

#### Example: Count



Counter word CW3 is incremented by 1 and/or decremented by 1.

A.4 Timer (T...)

Timers enable you to program switch-on and switch-off delays and other time functions (e.g. runtime monitoring). There are 256 timers available. Three operands are assigned to each timer. They provide information on the timer status, the current runtime and the pre-selected runtime (preset value) of the timer.

Operand	Designation	Description
T0 ... T255 TON0 ... TON255 <sup>1)</sup> TOFF0 ... TOFF255 <sup>1)</sup>	Timer status  (Pulse timer, switch-on or switch-off delayed timer)	Single-bit operand that indicates whether a timer is active or inactive (see also Tab. A/48). – If Timer (Tnn) <sup>2)</sup> is active, the corresponding timer presetting is loaded to the timer word before the timer is started. This applies also if the timer was already active. – When the timer (Tnn) is deactivated, it is halted. The timer word is automatically reset to 0. – The timer (Tnn) is automatically deactivated when the timer word has reached 0.
TP0 ... TP255	Timer pre-selection	Multi-operand, which contains the runtime of the timer in 0.01 s increments. Permitted runtime: 0.00s ... 655.35s <sup>3)</sup>
TW0 ... TW255	Timer word	Multi-bit operand that contains the current runtime of the timer.
<sup>1)</sup> Valid only in LDR <sup>2)</sup> nn represents the address of the counter (0...255). <sup>3)</sup> Instead of an absolute time value in seconds, the content of any multi-bit operand (e.g. IW0) can be loaded to the timer preset. Its content is automatically multiplied by a cycle time of 0.01 s and the result used as a timer preset.		

Tab. A/47: Timer operands



### Types of timer

In STL, you can use timers as pulse timers. In LDR, timers can also be used directly as switch-on and switch-off delay timers. The type of timer is determined in LDR by using the relevant operand for the timer status.

Types of timer	Timer status			Permitted in
	Operand	Active	Inactive	
Pulse timer	T0 ... T255	1	0	STL and LDR
Switch-on delay timer	TON0 ... TON255	0	1	LDR
Switch-off delay timer	TOFF0 ... TOFF255	1	0	LDR

Tab. A/48: Operands for the timer status

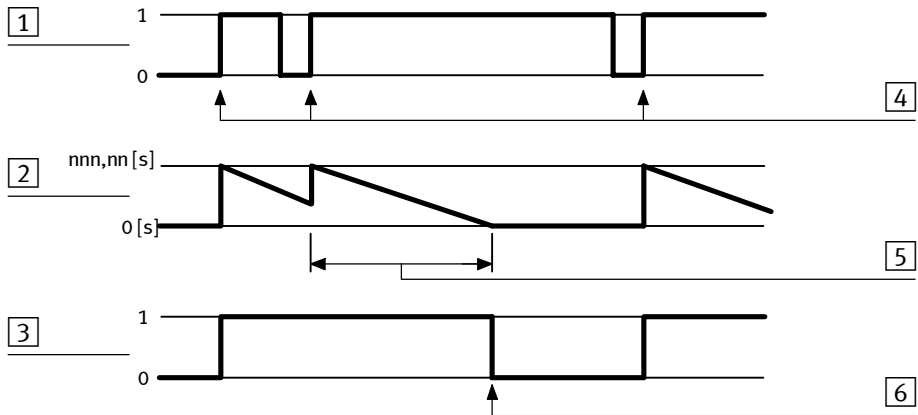
The timer status is a single-bit operand that can be set, reset or requested.

## Pulse timer

### Using timers as pulse timers

The pulse timer is started by a positive edge ( $Tnn=1$ ). The timer preset ( $TPnn$ ) is automatically loaded to the timer word and the timer starts to run. The timer word is decremented until:

- it reaches the value 0. The timer has then expired ( $Tnn=0$ )
- another positive edge (pulse) is present in the condition part, which causes the timer to restart.
- the timer status is reset (timer halted).



- |                           |                                 |
|---------------------------|---------------------------------|
| 1 Status of the condition | 4 Restart through positive edge |
| 2 Timer word              | 5 Timer preset                  |
| 3 Timer status Tnn        | 6 Timer expired                 |

Fig. A/10: Behaviour of pulse timers (LDR and STL)

### STL

Before a timer can be used, its timer preset must be initialised by a value consistent with the required time period. This initialisation only has to be repeated if the time value is changed. The timer preset must be reloaded at every timer restart. Timer presets can be loaded either with a constant or with the content of a multi-bit operand (e.g. Register, input word, flag word etc.).

#### Example: Pulse timer

```
""Alternative 1
...
IF          I0.1
    AND     N    T7      "Timer inactive?
THEN LOAD   V520      "5.2 s
    TO      TP7      "Load timer preset
    SET     T7      "Start timer
...
""Alternative 2
...
IF          I0.1
    AND     N    T7
THEN SET    T7
    WITH    5.2s
...
```

The LOAD... TO operation loads the timer preset. Once set, the timer is activated and starts to run.

#### Example: Requesting pulse timer

```
...
IF          N    T7
THEN RESET  O0.0
...
```

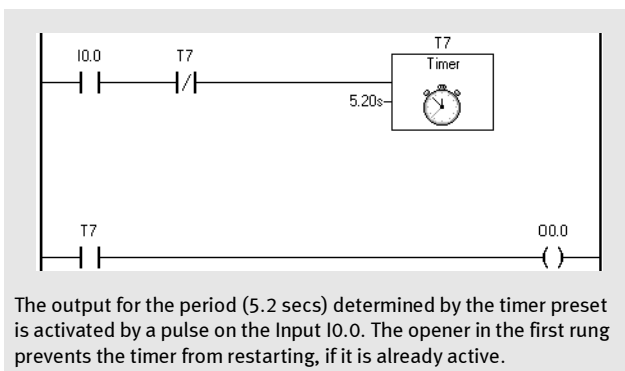
When timer T7 has expired, output O0.0 is reset.

## LDR

The timer is initialised by the timer box. For a pulse timer, enter an operand for the timer status (Tnn). When the link result has an rising edge, the timer box carries out the following operations:

- The value of the source operand located at the input (left) is loaded to the timer preset (TPnn) of the above timer.
- The timer is activated (timer status (Tnn) is set.

### Example: Pulse timer



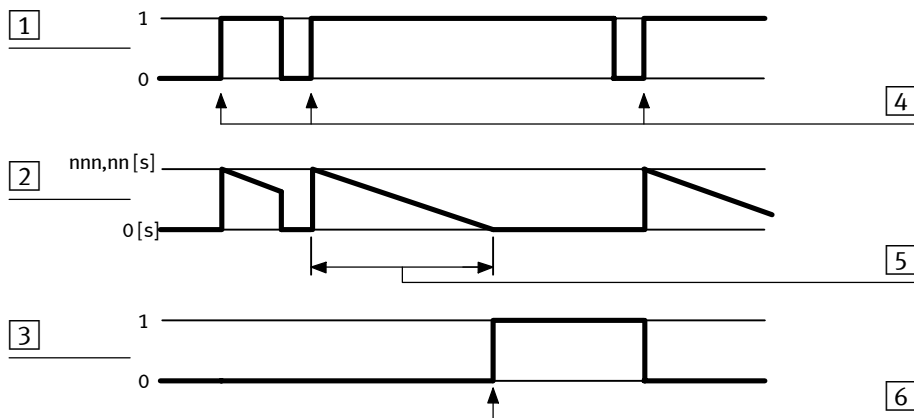
The output for the period (5.2 secs) determined by the timer preset is activated by a pulse on the Input I0.0. The opener in the first rung prevents the timer from restarting, if it is already active.

## Switch-on delay timer

### Using timers as switch-on delay timers

In LDR programs, you can use switch-on delay timers to activate outputs by a 1 signal and expiry of a delay time. The timer preset is the delay time. While the condition part carries 0 signal, the timer preset is permanently loaded to the timer word. At 1-signal, the timer starts to run until

- the timer word reaches the value 0. The timer has then expired (TONnn=1, TWnn=0).
- the timer is re-initialised by a 0-signal.



1 Status of the condition

2 Timer word

3 Timer status TONnn

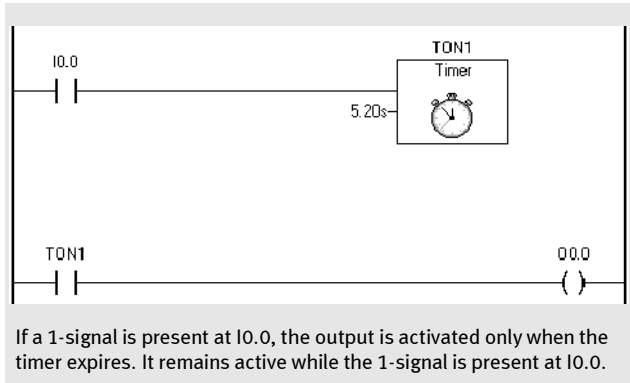
4 Restart through positive edge

5 Timer preset

6 Timer expired

Fig. A/11: Behaviour of switch-on delay timers (LDR only)

### Example: Switch-on delay timer



### Switch-off delay timer

#### Using timers as switch-off delay timers

In LDR programs, you can use switch-off delay timers to deactivate outputs by a 0 signal and expiry of a delay time. The timer preset is the delay time. While the condition part carries 1 signal, the timer preset is permanently loaded to the timer word. At 0-signal, the timer starts to run until

- the timer word reaches the value 0. The timer has then expired (TOFFnn=0, TWnn=0).
- the timer is re-initialised by a 1-signal.

## A. Operations (STL and LDR)

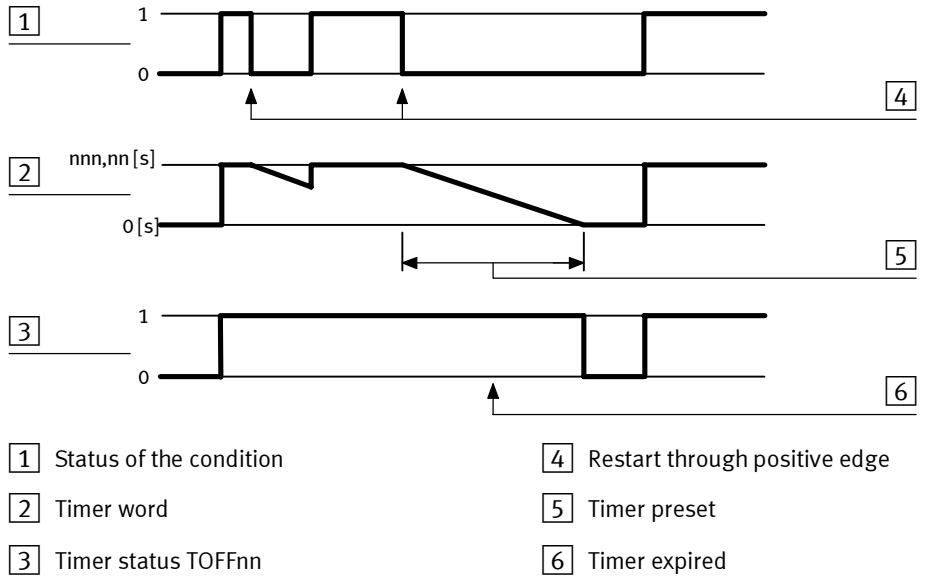
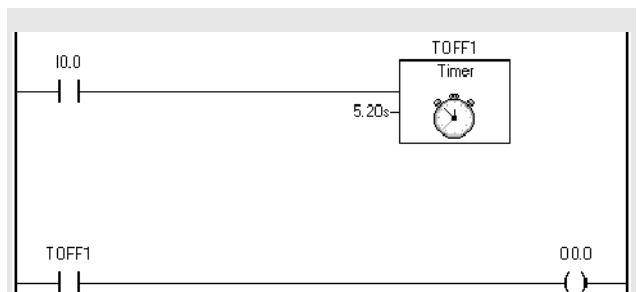


Fig. A/12: Behaviour of switch-off delay timers (LDR only)

### Example: Switch-off delay timer



The timer is started when a 0-signal is present at I0.0. TOFF1 and thus also output O0.0 run until the timer has expired, provided 1 signal is present.

A.5 Module call (CMP, CFM)

Modules are sub-programs that can be created in STL, LDR and C. A distinction is drawn between program modules (CMP) and function modules (CFM). They differ as follows:

Property	CMP	CFM
Step structure	Permitted	Not permitted
Task change at selection	Yes	No
Call of another CFM	Permitted	Permitted
Call of another CMP	Not permitted	Not permitted

Tab. A/49: Differences between CMP/CFM



The possible nest depth depends on the available memory. It is recommended not to exceed a nest depth of 10.

During the access process, program and function modules can be provided with up to 7 input parameters and return up to 7 feedback parameters to the selecting program. The input and feedback parameters are transferred to the local program function units FU32 to FU38.



The runtime library supplied with FST contains several pre-prepared modules, most of which are created in C, which can often be used only in combination with certain drivers. For these modules, the precise parameter transfer specification can be found in Volume 2.

When a module is created or imported into a project, the module is assigned a number (0...99). The module is selected in the project via this number.



## CMP, CFM

### Calling module

#### STL

Program modules are called with CMP, function modules with CFM. Up to 7 transfer and feedback parameters are possible. The parameters are transferred via the local program function units FU32 to FU38.

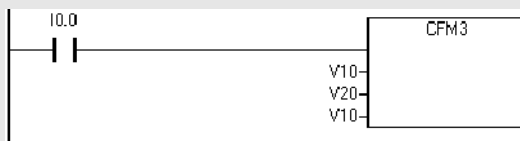
#### Example of CMP call

IF		I1.2	
THEN	CMP 3		"Copy flag word area
	WITH	V10	"No. of first source
			"flag word (FU32)
	WITH	V20	"No. of first target
			"flag word (FU33)
	WITH	V10	"No. of flag words
			" (FU34)

#### LDR

A module is called using the module symbol. The module is selected with an rising edge of the link result.

#### Example



With a rising edge to I0.0, CMP 3 is called. When called, 3 values (10, 20, 10) are transferred as the input parameters.

A.6 Jump (JMP TO)

**JMP TO** Jump to

The jump command cancels the program run at the current location and sets it for continuation at the indicated position within the current program.

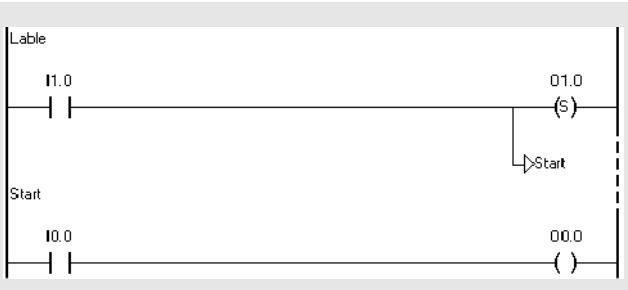
STL In STL, the JMP TO instruction executes a jump to a specific step. A maximum of 255 steps per program are possible.

**Example**

```
STEP label
  IF          I1.0
  THEN SET    O1.0
  JMP TO Start
...
STEP Start
...
```

LDR Each rung can be tagged with a jump label. A maximum of 255 jump labels per program are possible. The jump symbol is an arrow. The jump destination is indicated next to it.

**Example**



# **Menu commands**

## **Appendix B**

Contents

**B.      Menu commands ..... B-1**

B.1      Commands in the menu bar ..... B-3

## B.1 Commands in the menu bar

### Commands in [Project] menu



Commands in [Project] menu	
[New...]	Creates a new project. The current project will be closed after a request to save (see Section 2.2.1).
[Open...]	Opens a saved project (see Section 2.2.3).
[Close]	Closes the current project. If the project is re-created or changed, a prompt appears asking whether the project is to be saved or not (see Section 2.2.4).
[Settings...]	Opens the Project settings dialog window (see Section 2.2.2).
[Make Project]	Compiles only programs that were changed since the compilation (see Section 2.2.6).
[Build Project]	Compiles all programs, regardless of whether they were changed (see Section 2.2.6).
[Clean Up]	Deletes unrequired temporary files in the project in order to create memory on the data carrier (see Section 2.2.11).
[List Project File]	Displays the content of the last project file to be created (see Section 2.2.10).
[Explore]	Opens manage projects dialog window, enabling you to conveniently delete, copy and rename projects (see Section 2.2.5).
[Backup...]	Creates a ZIP file from the current project (see Section 2.2.14).
[Restore...]	Restores an archived project (see Section 2.2.14).
[Print...]	Opens a dialog for selecting the project parts that are to be printed (see Section 2.2.13).

## B. Menu commands

Commands in [Project] menu	
[Send Mail]	If you have an e-mail application and a mail server, you can send from FST an e-mail with a FST project as an attachment. FST uses the MAPI interface in Windows to communicate with the e-mail application.
[1 ...] [2 ...] [3 ...] ...	The last opened projects are shown as menu commands and can be opened directly.
[Exit]	Ends FST

## Commands in the [Edit] menu

Commands in the [Edit] menu	
[Undo]	Undoes the last changes to a document in the usual manner (see Section 3.2.2 and 4.2.3).
[Redo]	Restores the original status (see Section 3.2.2 and 4.2.3).
[Cut]	Deletes selected objects from the current window and places them on the clipboard. The previous clipboard content is lost.
[Copy]	Copies selected objects from the current document to the clipboard.
[Paste]	Inserts the contents of the clipboard to the current document at the selected position.
[Delete]	Deletes selected objects from the current document. The data are <b>not</b> copied to the clipboard.
[Find...]	Opens the "Find..." dialog window that enables you to find text in the current document (see Section 3.2.2 and 4.2.3).





## B. Menu commands




Commands in the [Edit] menu	
[Global Find...]	<p>Opens the “Global Find...” dialog that enables you to find text or operands in all programs throughout the entire project. Only the program source texts are searched. The automatically-generated comments are not part of the program source text.</p> <ul style="list-style-type: none"><li>– The “Find operands only” option finds symbolic and absolute operands, regardless of whether the search item was the symbolic or the absolute designation.</li><li>– The “Only loaded programs” option searches through only the programs selected for loading.</li></ul>
[Find next]	<p>Repeats the last search (see Section 4.2.3 and 3.2.2).</p>
[Replace...]	<p>Opens the “Replace...” dialog that enables you to replace text in the current document (see Section 3.2.2 and 4.2.3).</p>
[Select All]	<p>Selects the entire content of the document.</p>
<p><sup>1)</sup> The command is active if no item is selected. <sup>2)</sup> The command is inactive if the clipboard is empty.</p>	

### Commands in the [View] menu


 Allocation List

 Strings

 Controller Settings

 Driver Configuration

 IO Configuration

 Project Documentation

Commands in the [View] menu	
[Allocation List]	Opens the allocation list of the current project (see section 2.6).
[Strings]	Opens the strings editor (see Section 2.7).
[Controller Settings]	Opens the “Controller Settings” dialog that enables you to make settings for the current project (see Section 2.5).
[Driver Configuration]	Opens the driver configuration (see Section 2.4).
[IO Configuration]	Opens the hardware configurator (see Section 2.3).
[Project Documentation]	Opens the PROJECT.TXT file in the current project directory (see Section 2.2.12).
[Toolbar]	Hides or shows the icon list (see Section 1.4.1).
[Status Bar]	Hides or shows the status line (see Section 1.4.1).
[Shortcuts]	Hides or shows the shortcut (see Section 3.2.2 and 4.2.3).
[Catalog]	Hides or shows the catalogue of the hardware configurator (see Section 2.3.2).
[Message Window]	Hides or shows the message window (see Section Fig. 2/9).
[Project Window]	Hides or shows the project window (see Section Fig. 2/3).



## Commands in the [Insert] menu

<b>Commands in the [Insert] menu</b>	
[New Program...]	Opens the “New Program...” dialog that enables you to create a new program (see Section 2.8.1).
[New CMP...]	Opens the “New Program...” dialog that enables you to create a new program module (see Section 2.8.1).
[New CFM...]	Opens the “New Program...” dialog that enables you to create a new function module (see Section 2.8.1).
[STL Instruction]	Offers STL instructions for inserting into the current STL program (see Section 3.2.2).
[LDR Element]	Offers LDR elements for inserting into the current LDR program (see Section 4.2.3).
[Module Call...]	Enables a module call to be inserted into a STL program via the dialog (see Fig. 3/7).
[Operand...]	Opens a dialog that enables you to select an operand from the allocation list and insert it into the current program (see Fig. 3/5 and Fig. 4/10).
[Driver...]	Opens the select drivers dialog (see Fig. 2/32).
[IO Module...]	Opens the “IO Module Entry” dialog (see Fig. 2/18).

## Commands in the [Program] menu

<b>Commands in the [Program] menu</b>	
[New...]	Opens the “New Program” dialog that enables you to create a new program (see Section 2.8.1).
[Open...]	Opens the “Open Program” dialog that enables you to open new program saved in the project (see Section 2.8.6).
[Save]	Saves the current project (see also Section 2.8)
[Save As...]	Saves the current program under a different program or version number (to create a copy of the current program, see also Section 2.8).
[Save All]	Saves all changed documents.
[Import...]	Enables programs and modules to be imported into the current project (see Section 2.8.2).
[Export...]	Enables modules to be exported to the library so that they can be used in other projects at a later time (see Section 2.8.3).
[Delete...]	Opens the “Delete Program” dialog that enables you to delete a program saved in the project (see Section 2.8.5).
[Select for Download]	Enables you to select the programs that are to be downloaded when the project is imported (see Section 2.8.8).
[Print...]	Enables the current program source to be printed in the active window (see also Section 2.2.13).
[Send Mail]	If you have an e-mail application and a mail server, you can send from FST an e-mail with a FST program as an attachment. FST uses the MAPI interface in Windows to communicate with the e-mail application.
[Compile]	Compiles the current project into the internal command syntax (machine code) of the controller (see Section 2.8.7).



## Commands in [Online] menu



Commands in the [Online] menu	
[Login]	Sets up an online link to the connected controller in order to test the link and displays the following information (see Fig. 2/54): <ul style="list-style-type: none"> <li>– Controller type</li> <li>– Operating system version (basic unit)</li> <li>– Name of the active project</li> </ul>
[Refresh]	Upload again configuration and parameterisation for CPX terminal (see Section 2.3.8).
[Logout]	Ends an online link to the connected controller, closes all online windows and re-enables any serial ports that have been used.
[Control Panel]	Opens the online control panel that offers access to important basic functions and information on start-up and diagnosis (see Section 2.9.2).
[Terminal]	Opens the command interpreter terminal that enables you to send CI commands to the connected controller (see Section 2.9.6).
[Online Display]	Opens the online display that shows the operands of the connected controller. You change change operands and force inputs and outputs (see Section 2.9.4).
[File Transfer]	Opens a dialog that enables you to download files to the controller or upload from the controller and delete (see Section 2.9.5).
[IO Configuration]	Opens the hardware configurator in online mode (see Section 2.3.2).
[Web Browser]	Opens the integrated web browser, which displays the homepage of the connected controller (see Section 2.9.9).
[Download Project]	Downloads the current project to the connected controller (see Section 2.2.7).



<b>Commands in the [Online] menu</b>	
[Update Project]	Downloads to the controller only those changes made since the last download operation (see Section 2.2.8).
[Upload Project]	Restores a project from the sources stored in the controller (see Section 2.2.9).
[Editor]	Switches the current window to editor mode. This is supported by the following windows: <ul style="list-style-type: none"> <li>– STL editor window (see Section 3.5.2)</li> <li>– LDR editor window (see Section 4.4.2).</li> <li>– Hardware configur. CPX (see Section 2.3.3)</li> </ul>
[Online]	Switches the current window to online mode. This is supported by the following windows: <ul style="list-style-type: none"> <li>– STL editor window (see Section 3.5)</li> <li>– LDR editor window (see Section 4.4).</li> <li>– Hardware configur. CPX (see Section 2.3.3)</li> </ul>
[Breakpoints...]	Displays all the breakpoints set in the project in the form of an overview (see Fig. 3/11).
[Signed]	Switches the display format for the online display of multi-bit values to: <ul style="list-style-type: none"> <li>– decimal with sign (see Section 2.9.4).</li> </ul>
[Unsigned]	Switches the display format for the online display of multi-bit values to: <ul style="list-style-type: none"> <li>– decimal without sign (see Section 2.9.4).</li> </ul>
[Hexadecimal]	Switches the display format for the online display of multi-bit values to: <ul style="list-style-type: none"> <li>– hexadecimal (see Section 2.9.4).</li> </ul>
[Goto...]	Enables you to scroll directly to a desired step or path in the STL or LDR online display.
[Change Update Speed...]	Enables you to view and modify the delay time between the individual online requests (see Fig. 2/59).

## Commands in the [Extras] menu

<b>Commands in the [Extras] menu</b>	
[Preferences...]	Opens the “Preferences...” dialog that enables you to make the following settings: <ul style="list-style-type: none"> <li>– General settings for FST (see Fig. 2/2)</li> <li>– Settings for communicating with the controller (see Fig. 2/52)</li> <li>– Settings for STL and LDR editor (see Fig. 3/2 and Fig. 4/3)</li> <li>– Printer settings (see Fig. 2/16).</li> </ul>
[Library...]	Opens the runtime library that enables you to manage the drivers and I/O scripts (see Section 2.10).
[Convert FST 3.x STL Projects]	Starts an external tool for converting FSTIPC3.x projects into FST4.x projects
[AS-Interface Configuration]	Starts the AS-Interface configurator (see also Volume 2)
[Festo Fieldbus Configuration]	Starts the Festo fieldbus configurator (see also Volume 2)
[FED Designer]	Starts the FED Designer (the FED Designer must be installed separately).
[Configure Tools...]	Opens a dialog for managing external tools to be started via FST (see Section 2.11).

## Commands in the [Window] menu

<b>Commands in the [Window] menu</b>	
[Close]	Closes the active document and/or program window. If the document contains unsaved changes in the active window, the relevant prompt will appear. You can also close FST windows using the “Close” button in the title bar of the corresponding window.
[Cascade]	Cascades the document windows behind each other.
[Tile Horizontal]	Tiles the document window equally from top to bottom.
[Tile Vertical]	Tiles the document window equally from left to right.
[Arrange Icons]	Minimises the windows to icons at the lower edge of the FST program window.
[1 ..., 2 ..., 3 ...]	A list of names for the opened windows is shown in the bottom section of the menu [Window]. A tick indicates which window is active. Clicking on a window name activates that particular window.

## Commands in the [Help] menu

<b>Commands in the [Help] menu</b>	
[Help Topics]	Opens the online help for FST (see Section 1.4.5).
[Tip of the Day]	Displays a tip on the current program version of FST (see also Section 1.4).
[Internet -> ]	Offers internet connections for selection. If you have an Internet connection, you can access online information on Festo on the Internet.
[About FST]	Displays program information, version numbers of FST and copyright.

# **Index**

## **Appendix C**

Contents

C.      **Index** ..... **C-1**

C.1     Index ..... C-3



## C.1 Index

### A

#### Abbreviations

Product-specific .....	XXIII
Absolute operands .....	2-70
Activate password protection for online access .....	2-96
Actual-Nominal-Comparison .....	2-41
Allocation list .....	2-69
Automatic allocation list entry .....	3-11, 4-15
Changing allocation list entry .....	2-73
Comment .....	2-73
Deleting an allocation list entry .....	2-74
Editing regulations .....	2-72
Print .....	2-74
AND .....	3-27, 3-30, 4-13, 4-24, A-11, A-22
Archive	
Project .....	2-23, 2-30
Archiving .....	2-30
ASSIGNMENT .....	4-21, A-7
Autostart .....	2-67

### B

Backup .....	2-30
Backwards counter .....	A-41
BID .....	3-33, 4-23, A-28
Boxes .....	4-22
Breakpoint list .....	3-45
Breakpoints .....	3-41, 4-29

## C

Catalog .....	2-44
CFM .....	3-37, A-52, A-53
Change operand value .....	2-103
Change password in the controller .....	2-98
Change programm call .....	2-125
Change update speed .....	2-100
Changing address mapping .....	2-48
Changing parameters .....	2-52
Checking configuration .....	2-46
CI .....	5-31
CI Commands .....	5-35
Break (B) .....	5-46
Command letter .....	5-36
Command structure .....	5-35
Display (D) .....	5-39
Driver-specific commands .....	5-52
Init (Y!) .....	5-50
Linking .....	5-53
Modify (M) .....	5-43
Parameters .....	5-36
Run (R) .....	5-46
Stop (S) .....	5-47
Clipboard .....	3-16, 4-17
Close	
Close project .....	2-17
CMP .....	3-38, A-52, A-53
Coil .....	4-21
Command Interpreter .....	2-113, 5-31
Command interpreter	
Login .....	5-33
Comparison box .....	4-20

Compile	
Compile all .....	2-20
Module .....	2-89
Program .....	2-89
Project .....	2-20
Compiling	
Program and module .....	2-89
Condition part .....	3-20
Configure online link .....	2-92
Constants .....	5-12
Contact .....	4-19
Context menus .....	1-15
Controller COM Port .....	2-64
Copy	
Hardware configuration .....	2-31
Project .....	2-18
Counter .....	4-25
Counter box .....	4-11
Counters .....	5-11, A-37
CPL .....	3-34, 4-23, A-35
Create	
Program .....	2-79
Create backup copy .....	3-19

## D

D .....	A-20
De-archive	
Project .....	2-30
De-archiving .....	2-30
De-install drivers .....	2-123
Deactivate password protection for online access .....	2-96
DEB .....	3-34, 4-13, 4-23, A-30
DEC .....	3-35, A-20

DECREMENT .....	4-21
Deinstall	
FST .....	1-6
I/O script .....	2-120
Deinstalling FST .....	1-6
Delete	
Driver .....	2-59
File .....	2-112
Files from the controller .....	2-109
Module .....	2-86
Program .....	2-86
Programs and modules .....	2-86
Project .....	2-18
Delete project before download .....	2-67
Deleting module .....	2-46
Designated use .....	XV
Documentation .....	2-25
Download	
File .....	2-112
Files to the controller .....	2-109
Project .....	2-21
Sources .....	2-23
Download modified driver files .....	2-68
Download source files .....	2-68
Driver	
Configuration .....	2-55
Number .....	2-55
Open configurator .....	2-56
Properties .....	2-58
Drivers .....	2-62, 2-117
<b>E</b>	
Error handling .....	5-24
With error program .....	5-26
Without error program .....	5-25
Error output .....	2-62

Error program .....	2-62 , 5-26
Error status and error word .....	5-14
Executive part .....	3-20 , 3-21 , 3-24
EXOR .....	3-28 , 3-30 , 4-24 , A-13 , A-25
Explore	
Project .....	2-18
Export	
Program .....	2-84
Programs and modules .....	2-84
External tools .....	2-124
Extract	
Drivers .....	2-122
I/O script .....	2-120

## **F**

Fault number .....	5-28
Festo fieldbus .....	2-101
File transfer .....	2-109
Find and replace .....	3-17 , 4-17
Flag .....	5-8
Forcing .....	2-105 , 2-107 , 5-48
Fowards counter .....	A-38
FST installation .....	1-4
FST Kernel .....	2-63
Function modules .....	2-78
Function units .....	5-12

## **G**

Global find .....	3-18
Globally change	
Force values .....	2-108

Globally deleting	
Force values .....	2-107

## H

Hardware configuration	
Actual-Nominal-Comparison .....	2-41
Changing address mapping .....	2-48
Changing entry .....	2-35
Deleting an entry .....	2-35
For PS1, FEC Standard and FEC Compact .....	2-32
For the CPX terminal .....	2-36
Saving actual configuration as nominal configuration	2-39
Hardware configurator	
Catalog .....	2-44
Changing parameters .....	2-52
Checking configuration .....	2-46
Deleting module .....	2-46
Generating the nominal configuration .....	2-43
Inserting module .....	2-44
Resetting configuration .....	2-47
Help .....	1-19

## I

I .....	A-20
I/O scripts .....	2-117
Icon bar .....	1-12
IF .....	3-26
Import	
Module .....	2-81
Programs .....	2-81
Programs and modules .....	2-81
Important user instructions .....	XVIII
INC .....	3-35, A-20
INCREMENT .....	4-21
Index .....	5-8

Initial execution flag .....	5-15
Inputs .....	5-6
Insert	
Driver .....	2-57
Insert IO Module .....	2-33
Insert module call via dialog .....	3-13
Insert operand into allocation list .....	2-72
Insert program call .....	2-125
Inserting an operand .....	4-14
Inserting LDR elements .....	4-10
Inserting module .....	2-44
Inserting STL instructions .....	3-7
Install	
Drivers .....	2-122
FST .....	1-4
I/O script .....	2-119
Intended readership .....	XVII
INV .....	3-34, 4-23, A-33

## **J**

JMP TO .....	3-30, A-54
Jump .....	3-30, 4-26, A-54
Jump labels .....	3-30, 4-13, A-54

## **K**

Key functions .....	1-13
---------------------	------

## **L**

Ladder diagram .....	4-4
LDR	
Fundamental principles .....	4-4

Operations (LDR and STL) .....	A-1
Shortcut bar .....	4-10
LDR editor	
Functions .....	4-6
Preferences .....	4-7
LDR symbols .....	4-19
List project file .....	2-24
LOAD .....	3-29, A-8, A-16
LOAD TO .....	4-23, A-17

## M

Macros .....	2-127
Manage drivers in the runtime library .....	2-121
Manage I/O scripts in the runtime library .....	2-119
Manage runtime library .....	2-117
Memory	
For project files and drivers .....	5-55
Memory for project files and drivers .....	5-55
Working memory for programs and drivers .....	5-56
Menu bar .....	1-11
Menu commands	
[Extras] menu .....	B-11
[Help] menu .....	B-12
[Insert] menu .....	B-7
[Online] menu .....	B-9
[Program] menu .....	B-8
[Project] menu .....	B-3
[View] menu .....	B-6
[Window] menu .....	B-12
Menu [Edit] .....	B-4
Module	
Module calls .....	3-37
Module library .....	2-81
Module call .....	4-25, A-52
Module properties .....	2-51



Multi-bit operands .....	2-70
Multi-bit operations .....	4-22
Multi-bit operations with 2 operands .....	4-23
Multi-bit operations with 3 operands .....	4-13
Multi-tasking .....	5-17
Multiple-column list fields .....	1-18

## **N**

N .....	3-28, A-10
NEGATED ASSIGNMENT .....	4-21
Nested program selection .....	5-19
New compared to FST 4.02 .....	1-7
Nominal configuration .....	2-43
NOP .....	3-28, A-9
Notes on the use of this manual .....	XVII

## **O**

Online	
STL online display .....	3-39
Online control panel .....	2-95
Online display .....	2-99
Online Help .....	1-19
Online mode .....	2-91
Open	
Driver configurator .....	2-56
Hardware configurator .....	2-31
Open project .....	2-16
Program .....	2-88
Program or module .....	2-88
Open CI terminal .....	2-113, 2-115
Open online control panel .....	2-95

Open runtime library .....	2-118
Operand comment .....	4-14
Operands .....	5-4
Absolute .....	2-70
Change operand value .....	2-103
Changing or forcing operand values .....	3-40
Multi-bit .....	2-70
Retentive operands .....	5-15
Single-bit .....	2-70
Symbolic .....	2-69
Operations	
+, -, *, / .....	3-36, A-18
AND .....	3-27, 3-30, A-11, A-22
ASSIGNMENT .....	A-7
Backwards counter .....	A-41
BID .....	3-33, A-28
CFM .....	3-37
CMP .....	3-38
CMP/CFM .....	A-53
CPL .....	3-34, A-35
DEB .....	3-34, A-30
DEC .....	3-35
EXOR .....	3-28, 3-30, A-13, A-25
Fowards counter .....	A-38
INC .....	3-35
INC, DEC, I, D .....	A-20
INV .....	3-34, A-33
JMP TO .....	3-30, A-54
LOAD .....	3-29, A-8, A-16
LOAD TO .....	A-17
NEGATION .....	3-28, A-10
NOP .....	3-28, A-9
OR .....	3-27, 3-30, A-12, A-24
Pulse timer .....	A-46
RESET .....	3-29, A-5
ROL .....	3-32, A-32
ROR .....	3-33
SET .....	3-29, A-4
SHIFT .....	3-31, A-6
SHL .....	3-32, A-30
SHR .....	3-32, A-32
SWAP .....	3-31, A-26

Switch-off delay timer .....	A-50
Switch-on delay timer .....	A-49
TO .....	3-29, A-8, A-17
OR .....	3-27, 3-30, 4-24, A-12, A-24
OTHRW .....	3-26
Outputs .....	5-7

## P

Parallel logic program .....	3-21, 3-23
Password .....	2-65
Setting by CI .....	5-51
Password protection .....	2-65, 2-97
Pictograms .....	XIX
Picture scroll bars .....	1-14
Print	
Allocation list .....	2-74
Driver configuration .....	2-59
Font .....	2-29
Hardware configuration .....	2-35
Margin .....	2-29
Project .....	2-26
Project parts .....	2-26
STL program .....	3-19
Print format	
Settings .....	2-29
Program	
Programs .....	2-78
Reset programs .....	2-61
Program and program status .....	5-13
Program modules .....	2-78
Program properties .....	2-80
Program window .....	1-12
Project	
Clean Up .....	2-25
Close project .....	2-17

Compile all .....	2-20
Copy project .....	2-18
Create new project .....	2-12
Create project .....	2-12
Delete restorable files .....	2-25
Documentation .....	2-25
Explore projects .....	2-18
General instructions .....	2-9
Open project .....	2-16
Organisation on the hard drive .....	2-5
Prepare for downloading .....	2-20
Print .....	2-26
Project documentation .....	2-25
Project properties .....	2-15
Rename project .....	2-18
Settings .....	2-15, 2-85
Setup project .....	2-12
Project directory .....	2-6
Project file .....	2-63
Project properties .....	2-85
Project window .....	1-12, 2-10
Properties	
Driver properties .....	2-58
Properties of programs and modules .....	2-85
Pulse timer .....	A-46

## R

Refresh .....	2-53
Rename	
Project .....	2-18
RESET .....	3-29, 4-21, A-5
Reset outputs .....	2-62
Resetting configuration .....	2-47
Retentive operands .....	5-15
ROL .....	3-32, 4-23, A-32
ROR .....	3-33, 4-23, A-32

RS232 link .....	2-91
Run LED .....	5-23
Rung .....	4-4
Rung comment .....	4-13
Rung number .....	4-13
Runtime behaviour (Run Mode) .....	2-60
Runtime library .....	2-117

## S

Safety regulations .....	XV
Saving actual configuration as nominal configuration ..	2-39
Selecting a graphic element .....	4-8
Selecting in the LDR editor .....	4-8
Selecting with the keyboard .....	4-9
Selecting with the mouse .....	4-9
Send a CI command .....	2-114
Serial port .....	5-31
Service .....	XVII
SET .....	3-29, 4-21, A-4
Set display format .....	2-101
Set/change the password .....	2-96
Setting password through the online control panel ....	2-98
Settings	
For downloading .....	2-67
PLC settings .....	2-60
SHIFT .....	3-31, A-6
SHL .....	3-32, 4-23, A-30
Shortcut bar .....	4-10
SHR .....	3-32, 4-23, A-32
Single-bit operands .....	2-70

Special characters .....	2-77
Start/stop input .....	2-61
Start/stop switch .....	5-22
Starting FST .....	1-6
Startup batch .....	2-63
Status bar .....	1-13
STEP .....	3-25
Step program .....	3-21
STL	
Brief description of the STL instructions .....	3-25
Fundamental principles .....	3-4
Long comments .....	3-10
Online display .....	3-39
Operations (LDR and STL) .....	A-1
Short comments .....	3-10
STL sentence .....	3-20
STL editor .....	3-5
Edit functions .....	3-7
Preferences .....	3-6
STL shortcut .....	3-7
Stop program .....	2-61
Stop project before download .....	2-67
Storing the password in the project .....	2-66
Strings Editor .....	2-75
Special characters .....	2-77
SWAP .....	3-31, 4-23, A-26
Switch-off delay timer .....	A-50
Switch-on delay timer .....	A-49
Symbolic operands .....	2-69
Syntax test .....	2-20
System settings .....	2-50

## T

Task change .....	5-17
TCP/IP .....	5-32
TCP/IP link .....	2-91
Test online link .....	2-94
Text markings .....	XIX
THEN .....	3-26
Time behaviour	
Time behaviour at start of sequence ..	A-46 , A-49 , A-51
Timer .....	4-25 , 5-9 , A-44
Title bar .....	1-11
TO .....	3-29 , A-8 , A-17
Types of timers .....	A-45

## U

Undo changes .....	3-18 , 4-18
Update	
Project .....	2-22
Project file content .....	2-24
Upload	
File .....	2-111
Files from the controller .....	2-109
Project .....	2-23
Sources .....	2-23

## W

Window edge .....	1-14
WITH .....	3-38
Working memory	
For programs and drivers .....	5-56
Workspace .....	1-12

## Z

Zoom function .....	4-18
---------------------	------





## Conditions of use for "Electronic documentation"

### I. Protection rights and scope of use

The file of your choice is subject to safeguarding provisions. Festo or third parties have protection rights for this electronic documentation which Festo provides on portable data storage devices (diskettes, CD ROM, cartridge discs), as well as in Internet and/or Intranet, always referred to in the following as "electronic documentation". In so far as third parties have whole or partial right of access to this electronic documentation, Festo has the appropriate rights of use. Festo permits the user the use under the following conditions:

#### 1. Scope of use

- The user of the electronic documentation is allowed to use this documentation for his own, exclusively company-internal purposes on any number of machines within his business premises (location). This right of use includes exclusively the right to save the electronic documentation on the central processors (machines) used at the location.
- The electronic documentation may be printed out on a printer at the location of the user as often as desired, providing this printout is printed with or kept in a safe place together with these conditions of use and other user instructions.
- With the exception of the Festo Logo, the user has the right to use pictures and texts from the electronic documentation for creating his own machine and system documentation. The use of the Festo logo requires written consent from Festo. The user himself is responsible for ensuring that the pictures and texts used match the machine/system or the relevant product.
- Further uses are permitted within the following framework:  
Copying exclusively for use within the framework of machine and system documentation from electronic documents of all documented supplier components.  
Demonstrating to third parties exclusively under guarantee that no data material is stored wholly or partly in other networks or other data storage devices or can be reproduced there.  
Passing on printouts to third parties not covered by the regulation in item 3, as well as any processing or other use, is not permitted.

#### 2. Copyright note

Every "Electronic document" receives a copyright note. This note must be included in every copy and in every printout.  
Example: © 2003, Festo AG & Co. KG,  
D-73726 Esslingen, Germany

### 3. Transferring the authorization of use

The user can transfer his authorization of use in the scope of and with the limitations of the conditions in accordance with items 1 and 2 completely to a third party. The third party must be made explicitly aware of these conditions of use.

### II. Exporting the electronic documentation

When exporting the electronic documentation, the licence holder must observe the export regulations of the exporting country and those of the purchasing country.

### III. Guarantee

- Festo products are being further developed with regard to hardware and software. The hardware status and, where applicable, the software status of the product can be found on the type plate of the product. If the electronic documentation, in whatever form, is not supplied with the product, i.e. is not supplied on a data storage device (diskette, CD ROM, cartridge disc) as a delivery unit with the relevant product, Festo does not guarantee that the electronic documentation corresponds to every hardware and software status of the product. In this case, the printed documentation from Festo accompanying the product is alone decisive for ensuring that the hardware and software status of the product matches that of the electronic documentation.
- The information contained in this electronic documentation can be amended by Festo without prior notice and does not commit Festo in any way.

### IV. Liability/limitations of liability

- Festo provides this electronic documentation in order to assist the user in creating his machine and system documentation. In the case of electronic documentation which in the form of portable data storage devices (diskettes, CD ROM, cartridge discs) does not accompany a product, i.e. which are not supplied together with that product, Festo does not guarantee that the electronic documentation separately available / supplied matches the product actually used by the user. The latter applies particularly to extracts of the documents for the user's own documentation. The guarantee and liability for separately available / supplied portable data storage devices, i.e. with the exception of the electronic documenta-

tion provided in Internet/Intranet, is limited exclusively to proper duplication of the software, whereby Festo guarantees that in each case the relevant portable data storage device or software contains the latest status of the documentation. In respect of the electronic documentation in Internet/Intranet it is not guaranteed that this has the same version status as the last printed edition.

2. Furthermore, Festo cannot be held liable for the lack of economic success or for damage or claims by third parties resulting from the use of the documentation by the user, with the exception of claims arising from infringement of the protection rights of third parties concerning the use of the electronic documentation.

3. The limitations of liability as per paragraphs 1 and 2 do not apply if, in cases of intent or wanton negligence or the lack of warranted quality, liability is absolutely necessary. In such a case, the liability of Festo is limited to the damage recognizable by Festo when the concrete circumstances are made known.

### VI. Safety guidelines/documentation

Guarantee and liability claims in conformity with the regulations mentioned above (items III. and IV) can only be made if the user has observed the safety guidelines of the documentation in conjunction with the use of the machine and its safety guidelines. The user himself is responsible for ensuring that the electronic documentation, which is not supplied with the product, matches the product actually used by the user.