Concept
User Manual

Volume 3

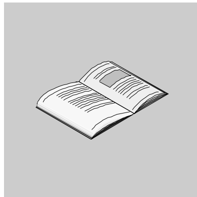
840 USE 503 00 eng Version 2.6 - SR1

a brand of Schneider Electric

33002204.01

Telemecanique

Table of Contents



The chapters marked gray are not included in this volume.

The chapters marked gray are not included in this volume.

About the Book

At a Glance

Document Scope

This user manual is intended to help you create a user program with Concept. It provides authoritative information on the individual program languages and on hardware configuration.

Validity Note

The documentation applies to Concept 2.6 for Microsoft Windows 98, Microsoft Windows 2000, Microsoft Windows XP and Microsoft Windows NT 4.x.

Note: Additional up-to-date tips can be found in the Concept README file.

Related Documents

Title of Documentation	Reference Number
Concept Installation Instructions	840 USE 502 00
Concept IEC Block Library	840 USE 504 00
Concept EFB User Manual	840 USE 505 00
Concept LL984 Block Library	840 USE 506 00

User Comments

We welcome your comments about this document. You can reach us by e-mail at TECHCOMM@modicon.com

Reference data editor

18

At a Glance

Overview

This Chapter describes the reference data editor (RDE) and its use with activated animation.

What's in this Chapter?

This chapter contains the following topics:

General Information about the Reference Data Editor

At a Glance

Variables can be displayed in animation mode, 0x and 1x references can be blocked (forced) and unlocated element variables or elements of structures can be set cyclically using the Reference Data Editor (RDE). The behavior of the variables can be followed and modified online through directly accessing the variables and direct addresses used in the IEC program. Variable states are displayed in animation mode with different colors (disabled, cyclically set).
Maximum 250 entries are possible in the Reference Data Editor. If this limit is exceeded a warning message is generated when saving.

Creating RDE Templates

To create an RDE template, use the variables declared in the variable editor. There are various possibilities here:

If ...	Then
You make a double click on the corresponding numerical field in the first column,	you open the dialog **Lookup variables**, for selecting a declared variable or component of a structure.
You enter the variable names of a declared variable in the column **Variable name**,	the declared parameters are entered into the RDE template.
You enter the direct address in the column **Address**,	then the value, the format and in some cases the defined name of the corresponding signal are entered in the RDE template.
You use menu command **Insert Addresses...** to insert entire reference blocks into the column **Address**,	the values and the formats of the corresponding signals are entered into the RDE template.

Display Signal States

Stored signal states are always overwritten by the current values in the PLC with an activated animation (**Online** → **Animation**) when opening an RDE template.
The signal states in the PLC can be displayed in online mode using menu instruction **Controller status...** . When starting the PLC, you can view signal states corresponding with the program progress in animation mode.

Printing RDE Templates

To print an open RDE template, click in the **RDE** main menu on the menu instruction **Print**. An exact copy of the screen image of the RDE template will be created on paper.

> **Note:** We recommend that you modify the printer properties to landscape paper format in the operating system (Windows). This will give you the complete image of the RDE template on a single page.

Using RDE Templates

Using an RDE template in more than one project is not recommended. This can cause doubled variable names to appear as well as variable names that did not exist in the original RDE template. The variables in the RDE templates are always displayed with the current reference addresses.

Converting RDE Templates

This procedure can be found in the description **Converting RDE Templates** (See *Converting RDE templates, p. 539*).

Converting RDE templates

Introduction

RDE templates from an earlier version of Concept are automatically converted into the template format of the new Concept version. To differentiate between the converted RDE templates and the other RDE templates, they are saved with the file extension *.RDF.

CAUTION

Incomplete RDE templates are created!

Before the conversion make sure that the variables in the RDE template are declared in the opened project in the new version of Concept. New variables are listed in an error message and cannot be displayed in the RDE template (*.RDF) created from it.

Failure to follow this precaution can result in injury or equipment damage.

Automatic Conversion

Automatic Conversion is performed when the RDE template of a previous version of Concept is opened:

Step	Action
1	Start the new version of Concept and open the project.
2	In the **Online** main menu click on the **Reference data editor** menu command. **Result:** The **RDE** main menu appears in the men bar.
3	In the **Online** main menu click on the **Reference data editor** menu command.
4	Select the directory, in which the RDE template (*.RDE) is saved (e.g. D:\CONCEPT_OLD). **Result:** All existing RDE templates (*.RDE or *.RDF) are displayed. **Note:** The files with the *.RDF extension come from the conversion of generated RDE templates (*.RDE).
5	Select the *RDE RDE template to be converted.
6	Click on the command button **OK**. **Result:** The **RDE AutoConvert** message appears. This informs you that the *RDE template was created in a previous version of Concept and is now being saved in a new format, so that it can be used in this version of Concept. The converted template is saved in a file with the *.RDF extension.
7	Click on the command button **OK**. **Result:** The converted RDE template (*.RDF) is displayed. **Warning:** All RDE template variables must be declared beforehand in the project. For new variables, the **RDE Template Errors** error message appears now, in which all faulty variables are listed. After closing the window, the converted RDE template opens, but only containing the declared variables.
8	Using the **Save reference data table under...** menu command, it is possible to save the converted RDE template in the directory in the new version of Concept (C:\CONCEPT_NEW). **Result:** The converted RDE template is stored in the Concept directory with the *.RDF file extension.

Changing signal states of a Located variable

At a Glance

Located variables can be changed by checking the corresponding signal box in the columnDisable and editing the value. Upon locking, the variable is separated from the hardware and is only used in the logic again if the disablement is undone. In this way, the changed signal states of all editors (FBD, SFC, LD, ST, LL984) are taken into account.

Forcing inputs and outputs

When inputs are forced, signal states are transferred until the value in the RDE table is changed again. When outputs are forced, the new value appears at the beginning of each program cycle. When a subsequent change is made using the program logic, this value is not saved in the state RAM until the locking of the output has been removed.

	CAUTION
	All changed signal states are loaded directly onto the PLC. Though not in the case of forced located variables. **Failure to follow this precaution can result in injury or equipment damage.**

Display of disabled variables

Variables that have been disabled by checking the check mark are shaded in color in the editor display. By removing the check symbol, the colored background of the corresponding variable is also no longer visible.

Loading reference data

Cyclically set values and disabled variables can be loaded onto the PLC using the menu command **Load reference data**.
These settings then remain the same until the user makes a change in the RDE table, or the PLC loses the loaded data (e.g. by loading a different project).

Note: In an open RDE table, the changed date is then automatically saved using the menu command **Load reference data**. The menu command **Save table** then no longer needs to be used.

Cyclical Setting of Variables

Introduction

Variables and structure elements can be changed by entering a set value corresponding to the data type of the variable in the **Set Value** column. This value will be written uniquely, if the corresponding signal’s box in the **Cyclic Set** column is subsequently checked. The new signal state is loaded directly onto the PLC and is transferred to the cyclically set variables administrator. The signal state of the variable, attained after logic editing at the end of the cycle, is specified in the **Value** column. In animation mode, the cyclical setting of variables in IEC sections is displayed.

Cyclic Set

> **Note:** Cyclical setting of variables can only be performed ONLINE and in EQUAL mode, not in animation mode. Depending on logic, the displayed value may deviate from the cyclically set value.

When the cyclical setting check box is checked, the set value in the **Set Value** column can still be changed.
If the box in the column **Cyclic Set** is unchecked, the signal state in the column **Value** is loaded onto the PLC and is used in the logic.
A maximum of 300 variables can be cyclically set. For cyclical setting, the length of the entry is limited to 150 characters in the column **Variable Name**, because this name is sent to control. If a variable is used several times in the reference data editor, the most recently entered value will always be the one taken into account for cyclical setting.

> **Note:** All changed signal states are loaded directly onto the PLC.

CAUTION

Modified variable names are not recognized by replacements.

When a variable is cyclically set, the spelling of the variable name should not be changed in the variables editor.

Failure to follow this precaution can result in injury or equipment damage.

Cyclical setting and locking of signal states in the operating modes:

Mode	Option	Meaning
LOCAL	Disable	The variables declared in the Variable Editor can be written in the RDE Template in local mode. The signal states specified in online mode are displayed in local mode but cannot be changed and have no effect.
ONLINE	Disable	The changed signal states of located variables are transferred directly from the program logic.
LOCAL	Cyclic Set	Cyclical setting of variables cannot be executed in local mode.
ONLINE	Cyclic Set	The signal state in the column **Set Value** is used in logic editing by checking the box (check mark visible), and supplies a value at the end of the cycle, which is displayed in the column **Value**.

Getting/deleting cyclical set list

The cyclical values set in animation mode can be inserted into the RDE Template in disabled animation using the menu command **Get CSL**.
Cyclically set values are recognized in the RDE Template by the check mark in the column **Cyclic Set**, and are automatically recognized row by row. It is therefore referred to as a cyclical set list. Using the menu command **Online** → **Get CSL** this recognized list will be inserted dependently from the selected row in the RDE table. Getting and inserting the cyclical set list can be done as often as required. The most recent cyclical set list is always located on the clipboard and can only be deleted using the menu command **Delete CSL**. Thereafter, getting and inserting is no longer possible until values are cyclically set at the next animation.

Note: Each time, the system gets **all** cyclically set values .

Downloading reference data

Cyclically set values and disabled variables can be loaded onto the PLC using the menu command **Download reference data**.
These settings then remain the same until the user makes a change in the RDE Template, or the PLC loses the loaded data (e.g. by loading a different project).

Note: In an open RDE Template, the changed date is then automatically saved using the menu command **Download reference data**. The menu command **Save template** then no longer needs to be used.

Unconditional locking of a section

At a Glance

At the section to be inhibited, the logic must carry a BOOL data type "output" and it should be noted that the section is disabled at configured "1".

	CAUTION
⚠	**Risk of unwanted process states.** Locking a section does not mean that programmed outputs within the section are deactivated. If an output was already set during a previous cycle, this state also remains after the section has been inhibited. It only ceases to be possible to change the state of these outputs once the section has been inhibited. **Failure to follow this precaution can result in injury or equipment damage.**

Note: A section that contains a logic to lock/release other sections should not be disabled, if possible. Output state disabled sections cannot be changed.

Procedure for unconditional locking of a section.

The following procedure is performed to disable a section unconditionally in the RDE table:

Step	Action
1	By double-clicking in a text box in the first column in the table (1 … 100) open the dialog box **Look up variables**.
2	In the zone **Data type** select the option button **Structured** and from the list select **SECT_CTRL**. **Reaction:** The names of all sections are displayed.
3	Select the name of the file to be disabled and using the command button **Elements...** open the dialog box **Select elements by type**.
4	Select the line **disable : BOOL** and confirm with **OK**. **Reaction:**The structured variable (Sectionname.disable) to which the section to be disabled is assigned, is entered in the RDE table.
5	Link the PLC and the programming device (**Online** → **Link...**), and load the user program onto the PLC (**Online** → **Load...**). **Reaction:** The PLC is in ONLINE and ANIMATIONS mode.
6	In the column **Value** enter a configured "1". **Reaction:** The section is disabled and will not be processed.

Animation

At a Glance

Animation can only take place in ONLINE mode. By activating Animation in the Reference data editor it is possible to display the signal states of the variables, and to observe the behavior of the output signals while the program is running.
During animation, signal states can be changed online also. The new values are automatically loaded onto the PLC and are taken into account during the next cycle.

Note: When changing a value it should be ensured that the locking of the variable is subsequently removed. It is impossible to animate disabled variables correctly.

Animation status

The column **Animation status** specifies the status of entered unlocated Variables during animation.
This table provides an overview of the animation status possibilities:

Display	Mode	Cause
Not used **Note:** In LOCAL mode, this display changes to "Unequal program"	ONLINE, ANIMATED	A variable not used in the user program, which is declared in the Variable Editor, was entered in the RDE table.
Inhibited I/O flag bits	ONLINE	An unlocated variable was cyclically set during the ANIMATIONS mode.
Unequal program	ONLINE	A variable that is used in the user program, which is declared in the Variable Editor, was entered in the RDE table. The program is in MODIFIED mode.
Unequal program Note: In ONLINE mode, this display changes to "Not used".	LOCAL	A variable not used in the user program, which is declared in the Variable Editor, was entered in the RDE table.

Display of forced and cyclically set signals in ANIMATIONS mode

The variables that are forced or cyclically set in the reference editor are labelled with a colored background in the individual editors.
Forced variables are displayed in the following way:

Editor	Display
IEC editors (FBD, LD, SFC, IL, ST)	When forcing occurs, variable names are shaded in ochre (brown-yellow).
LL984 editor	When forcing contacts, variable names are underlined. When forcing spools, an opened contact ("inhibited") is displayed before the spool.
Monitoring fields and **Display** dialog	When forcing occurs, variable names are shaded in ochre (brown-yellow).

Cyclically set variables are displayed in the following way:

Editor	Display
IEC editors (FBD, LD, SFC, IL, ST)	When cyclical setting occurs, the variable name is shaded in violet.
Monitoring fields and **Display** dialog	When cyclical setting occurs, the variable name is shaded in magenta.

Note: In LD (Ladder Diagram) spools and contacts are displayed in color. However, due to forcing and cyclical setting, it is possible that the colors of the variable names will be different from the color display of spools and contacts.

Display of forced and cyclically set element structured variables in ANIMATIONS mode

If a structured variable element is forced or cyclically set, there are different display possibilities.

Display	Cause
The name of the structured variable (e.g. motor) is shaded in color.	In the editor, a multi-element variable (e.g. motor) is displayed, in which one or more elements is forced or cyclically set.
The name of the structured variable element (e.g. right motor on) is shaded in color.	In the editor, a forced or cyclically set element of a multi-element variable (e.g. right motor on) is displayed.
The name of the structured variable element (e.g. right motor on) is shaded in color, but the name of the element is not.	In the editor, an element of a multi-element variable that is not forced or cyclically set is displayed, but a different element of this multi-element variable is cyclically set or forced.

Replacing variable names

At a Glance

When using an open RDE table it is possible to simultaneously edit the Variable Editor. If variable names are changed in the Variable Editor using the Find/replace function, these changes are automatically adopted in the open RDE table. In this case the RDE animation is initially terminated and the RDE table must be reloaded.

Procedure and reaction

For the automatic adoption of replaced variable names in the simultaneously open RDE table, the following steps are to be performed:

Step	Action
1	Open a section and create an online link. **Note:** The state between PLC and programming device must be EQUAL. If not, load the program into the PLC.
2	Start the animation (**Online** → **Animate binary values**). **Reaction:** The signal states of the section are displayed in color.
3	Open an existing RDE table (**RDE** → **Open table...**). **Reaction:** The RDE animation is started.
4	Open the Variable Editor (**Project** → **Variable declaration...**).
5	Using the command button **Find/replace** open the dialog **Find/replace**.
6	Replace an existing variable name with a new name (Command button **Replace**). **Reaction:** The variable name was changed in the Variable Editor.
7	Exit the Variable Editor using **OK**. **Reaction:** The section is automatically updated, and the RDE animation is terminated.
8	Close the RDE table and save the changes (Command button **Yes**).
9	Reopen the saved RDE table (**RDE** → **Open table...**). **Reaction:** The RDE animation with the changed variable name is recovered.

Load reference data

At a Glance

In the same cycle, the variables changed in the reference data editor are sent to the PLC, using the menu command **Online** → **Load reference data**.

Note: To perform the loading, the animation must be disabled.

ASCII Message Editor

19

At a glance

Introduction

This chapter describes the ASCII message editor.

What's in this Chapter?

This chapter contains the following sections:

19.1 ASCII Editor Dialog

At a glance

Introduction

This section describes the ASCII editor dialog.

What's in this Section?

This section contains the following topics:

Generals to ASCII editor dialog

Introduction

Use the ASCII message editor to create, edit, and simulate ASCII messages. The ASCII message text/control that is created in the editor can be transferred to the selected PLC. Conversely, the ASCII messages internal to the controller can be uploaded to the editor.

An ASCII message set consists only of a list of messages that satisfy certain rules. The number of messages allowed and the maximum length of the ASCII message set is defined as part of the PLC configuration. Each message consists of a list of ASCII message fields separated by commas.

The following fields are currently supported:

- *Text, p. 553*
- *Variables, p. 554*
- *Control code, p. 555*
- *Spaces, p. 555*
- *Carriage Return, p. 556*
- *Flush (buffer), p. 557*
- *Repeat, p. 558*

Preconditions

This function is only available when using:
- Concept for Quantum
- The modules J892 or P892
- Programming language LL984

Text

Introduction

The text messages defined by text fields take the format `'Hello World'` whereby `Hello World` becomes the text to be forwarded. The single quotation marks are the delimiters. The ASCII message editor development dialog provides a development area and a simulator area where the composed message is interpreted and displayed for you to make any necessary edits before leaving the editor dialog.

Message Length

An ASCII message can be as long as 134 words. Three words are for overhead plus the actual message maximum of 131 words (2 characters per word).
Message words are used up as follows:

Field type	Field length (in words)
ASCII text	1 + length of text / 2 rounded up
Return	1
Flush 0, 1	1
Flush 2, 3	2
Control	1
Variable	1
Repeat	2
Space	1

Variables

Introduction

A variable will be given the format NTF.

The meaning of this is:
- N representing the decimal number (1...99) of the data fields of the data type defined by T.
- T is the data type of the variable.
- F the decimal field width for the variable.

Data Types

The data types supported are:

Type	Repetition factor
A = ASCII character	1
B = binary number	1 to 16
H = hexadecimal	1 to 4
I = integer	1 to 8
L = integer with leading 0s	1 to 8
O = octal	1 to 6

Example

For example: 2H2 means:
- 2 registers (N)
- in hexadecimal (T)
- containing 2 hexadecimal numbers (F)

N can fit into the number of data registers needed, but it is not an absolute requirement.

The relationship is:

Type	Relationship
A	Number of registers = N/2 (next upper integer value)
B	Number of registers = N
H	for $1 \leq F \leq 4$... Number of registers = N for $5 \leq F \leq 8$... Number of registers = 2 x N
I and L	The same as H
O	Number of registers = N

Control code

Meaning of Control Code

A control code is given the format "Null", with Null being a three characters OOO, and the double quotation marks are delimiters.

For example: "017"

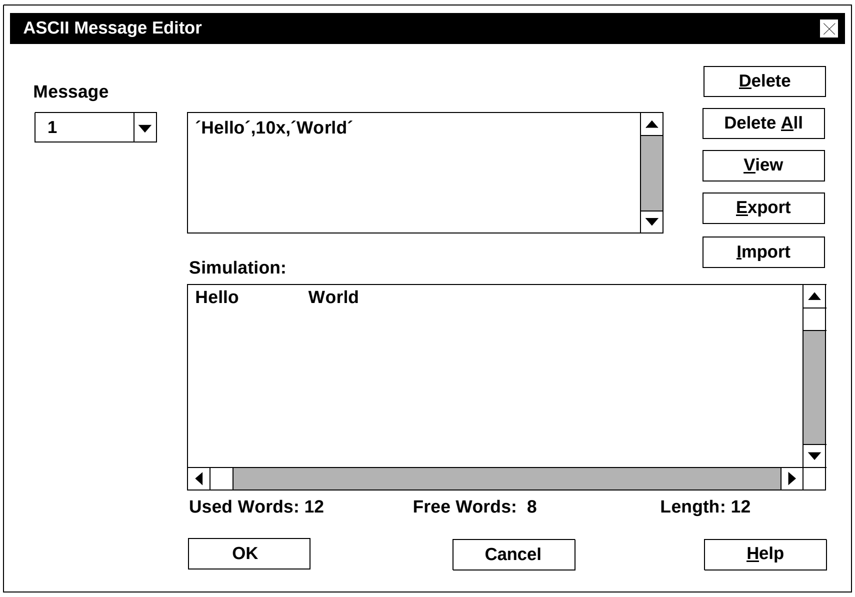
Spaces

Meaning of Spaces

A space field is given the format ddx, with dd being a decimal number (1..99) used to determine how many spaces are to be added to the message.

Representation of Dialog

Many spaces between text:



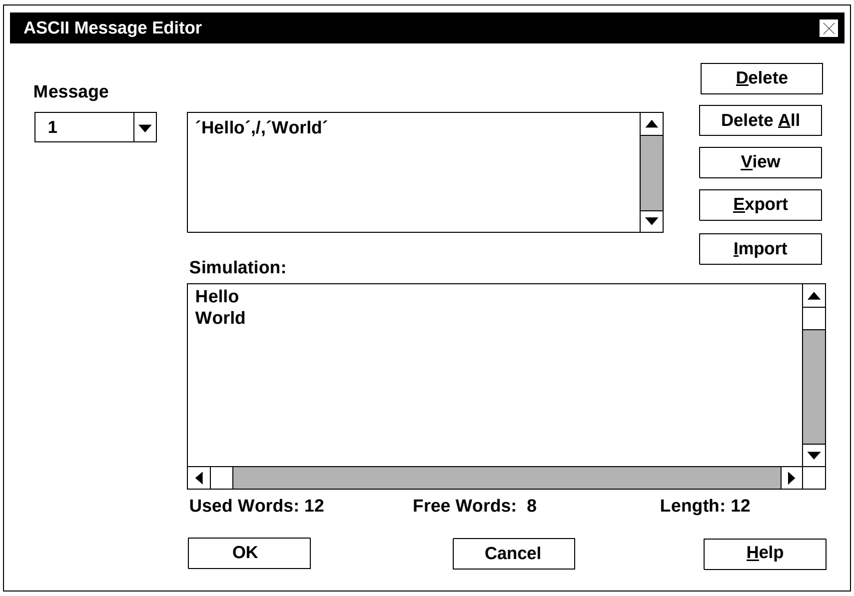
Carriage Return

Meaning of Carriage Return

A carriage return field will add a carriage return to the output information, and it has the format, /.

Representation of Dialog

Carriage return:



Flush (buffer)

Meaning of Flush

This will expressly specify for the P892 only how the input message buffer is to be cleared. This field has the format <*>/.
The * can be any of the following:

*	Meaning
0	Remove all characters in the buffer. An example is: <0> clears all.
1;bbb	Removing the number of characters specified by bbb, whereby bbb is a number (1...255). For example, <1;100> flushes the first 100 characters in a buffer.
2;hhhh	Scanning the message for the 2 characters that are specified by the hexadecimal numbers hhhh. If a match is found, delete all characters up to but not including the match. An example is: <2;5445> causes the buffer ‘12TEST’ to become "TEST".
3;rrr;hhhh	Scanning the message for the 2 characters that are specified by the hexadecimal numbers hhhh. If a match is found, delete all characters up to and including the match. The search is to be performed as often as specified by rrr, whereby rrr is representing a decimal number 1...255. **Example:** <3;2;5445> causes the buffer ‘12TEST3456TEST789TEST’ to become ST789TEST.

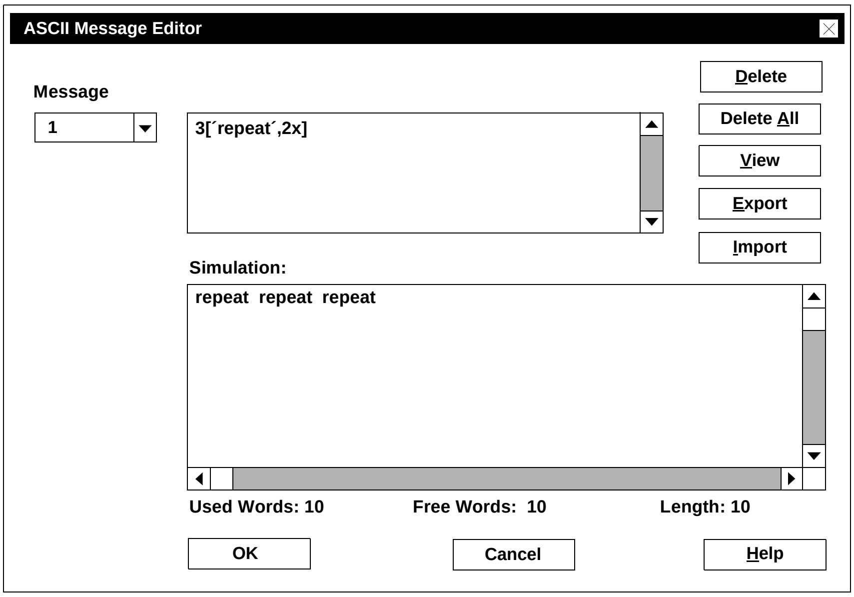
Repeat

Meaning of Repeat

Use this message field to specify that a number of message fields will be repeated several times. This field has the format dd(*), with dd being a decimal repetition factor (1....99), ( ) are delimiters, and * is a series of message fields.

Representation of Dialog

Repeated text:



19.2 User Interface of ASCII Message Editor

At a glance

Introduction

This section describes the user interface of the ASCII message editor.

What's in this Section?

This section contains the following topics:

How to Use the ASCII Message Editor

Invocation of ASCII Message Editor

The ASCII message editor is invoked from the **ASCII messages...** menu item in the **Project** menu. This editor allows you to add/modify/delete messages in a temporary work space, then save or cancel the changes.

Add New Messages

To add a new message, type the new message number into the **Message** text box and type a syntactically correct message into the message text box. As you enter a message into the message text box, its corresponding simulation is displayed in the **Simulation** text box. When the message is syntactically incorrect, it is displayed in red.

Modify Existing Messages

To modify an existing message, select a message from the **Message number** list and change the text.

Delete Messages

To delete a message, select a message from the Message number list and click on **Delete**.
Clicking on the button **Delete All** will remove all messages in the temporary workspace. The button is active if there is at least one ASCII message in the message set. Selecting this option results in the display of a confirmation dialog.

View

Clicking at the button **View** will produce a view of the displayed **ASCII message** dialog. The view message format is message number followed by message text. You can select from the choices available. To download the editor from the view list, click on the message and on **OK**.

Save Changes

Use the button **OK** to save processes performed while working with the ASCII editor and to close the dialog. Each message that has been created or changed is checked for syntactic correctness at this point. The checking begins at the current message and wraps around until all messages are checked. If a syntax error is detected, a definition of the error is displayed first, and as soon as the error dialog is cleared, the message itself appears with the cursor on the faulty character. Every attempt to add ASCII characters which will cause the size of the entire message area set in the configuration to be exceeded will generate an error.

Length, Used and Free

These fields display the length of the current message (in words), the number of words used and the number of words remaining.

Message Number

Introduction

The combo box **Message number** is a dialog that contains a message selection list with a check mark next to the currently selected message.

Use this dialog to select existing message numbers and/or to add new message numbers. As long as there are no messages, text box and list are empty. If there are messages, the editor is initially displayed with the text box containing the first message number and a list of message numbers for existing messages. The message number that relates to the currently displayed message is posted above the list box.

Action

For the selection of an existing message, click at the list button and mark a number in the list or type the number into the text field. A new message number can be inserted by typing the number into the text field.

Effects

If the message number assigned to an existing message is changed (either text or list entry), the text box **Message** will display the message text for the message number and the box **Simulation** shows the simulation of the message. If a new message number has been entered, the text boxes **Message** and **Simulation** will be cleared.

Error handling

The following errors can be appearing:

If...	Then ...
an unauthoried character is entered in the number field of the message.	a message field dialog will show: "Message number contains illegal characters". After acknowledging the error, the message number is reset and the process will continue in the text box **Message**.
the text box **Message** is not filled out.	a message field dialog will show: `"There must be a message number before text can be entered"`. After acknowledging the error, the message number is reset and the process will continue in the text box **Message**.
the number is greater than the maximum number set in **Configure** → **ASCII Setup...**.	a message field dialog will show: `" Message number exceeds maximum set in configuration"`. After acknowledging the error, the message number is reset and the process will continue in the text box **Message**.

Message Text

Introduction

The text box **Message** is a text editor with free format for the entry of ASCII messages. This editor allows one arbitrarily long line of free-format text. Although the text should follow the ASCII message syntax, it does not necessarily have to be syntactically correct prior to activating the **OK** button, even though a view note regarding validity will appear already during entry of the messages.

Actions

A currently selected message is made available for editing, otherwise a new message can be entered. The standard Windows edit operations (**Cut**, **Paste**, **Copy**, ...) are allowed.

Effects

If the message is syntactically correct, its text will be displayed in normal textual color, if not, the display will be in red. Text wraps so there is never a case where horizontal scrolling is required.

Simulation Text

Introduction

The text box **Simulation** is a read-only multi-line field. The simulated output of the current message is displayed in this window. As messages are added or changed, the simulated output is displayed in the simulation window.

Special Considerations

The simulation of control codes is shown as the ASCII character that corresponds to the controller, except those control codes that are not authorized in Windows text control and are written as an ’l’.

Note: Any simulation greater than 32 k characters is truncated to this maximum.

19.3 How to Continue after Getting a Warning

How to Continue after Getting a Warning

Introduction

A few conditions will allow continuing work with the ASCII editor although with possibly restricted functionality.

Note: To match a configuration, messages may be deleted.

Exceeding the Total Messages

Message numbers that are above the maximum limit set in **Configure** → **ASCII Setup...** will only be available for display or delete. These messages appear grayed out.
The accompanying warning reads: "`Warning: Some message numbers exceed the highest message number xx, defined in Configure. All messages beyond xx can only be displayed or deleted.`"

Exceeding the Message Area Size

If the size of the message in the data base is greater than the size defined in **Configure** → **ASCII Setup...**, a warning will appear. You can continue to view, change, or delete but changes cannot be saved unless the size falls below the configuration setting.
This warning reads: "`Warning: The size of the ASCII message area, xx, exceeds the maximum size, xx, defined in Configure.`"

Tips

Note: To match a configuration, messages may be deleted.

Note: Information about the ASCII character set can be found in the PLC User's Guide.

19.4 ASCII Editor in Offline/Combination/Direct Modes

ASCII Message Editor in Offline/Combination/Direct Modes

Offline

When using Concept to program in offline mode, the ASCII message editor is displayed with the set of messages saved in the data base. By activating the **OK** button, these messages will be saved in the database.

Direct

When using Concept to program in direct mode, the ASCII message editor will be displayed with the set of messages saved in the controller. By clicking on the **OK** button, the changes made to the ASCII messages will be downloaded to the controller.

Combination Mode

When entering the Combination mode, Concept checks whether the information in the controller matches the information in the data base. If a match is found, the controller is considered **EQUAL** to the database. A mismatch is marked as **NOT EQUAL**. If the status is **EQUAL**, the ASCII message editor will be displayed with the ASCII message set taken from the data base. If a displayed editor message is changed, these changes will be saved to the database and the controller after clicking the **OK** button.

Online functions

20

At a Glance

Overview

This chapter describes the various online functions.

What's in this Chapter?

This chapter contains the following sections:

20.1 General information about online functions

General information

At a Glance

After setting up a link via Modbus, Modbus Plus or TCP/IP between the programming device and the PLC the project can be loaded onto the PLC. Now special online functions for displaying and changing the current value in the PLC state RAM are available in the separate editors. The PLC can be controlled.

	CAUTION
⚠	**A communication timeout or a general memory protection failure could occur if the system clock of the programming device is changed while it is online.** If the running program cannot be terminated, all animated program sections should be closed , or the animation should be turned off in order to reduce the possibility of getting into a time critical operation. **Failure to follow this precaution can result in injury or equipment damage.**

20.2 Connect to PLC

At a Glance

Overview

This section contains information about connecting the PLC.

What's in this Section?

This section contains the following topics:

General

At a Glance

A connection can be created between a programming device and the PLC.
IEC sections can be modified in Monitor mode, they cannot however, be downloaded to the PLC. When exiting Concept a warning will be displayed.

> **Note:** Only one programming device may be connected the PLC.

Limited PLC Login

When logging into the PLC, the following restrictions are imposed for Quantum CPUs 140 434 12 A and 534 14 A:

- If a programming device is already connected with the PLC in programming mode, then no other programming devices can be connected with the PLC.
- If a programming device is already connected with the PLC in Monitor mode, then other programming devices can be only connected with the PLC in Monitor mode. An attempt to connect with the PLC in another operating mode is not possible for the other programming devices.

Consistency check

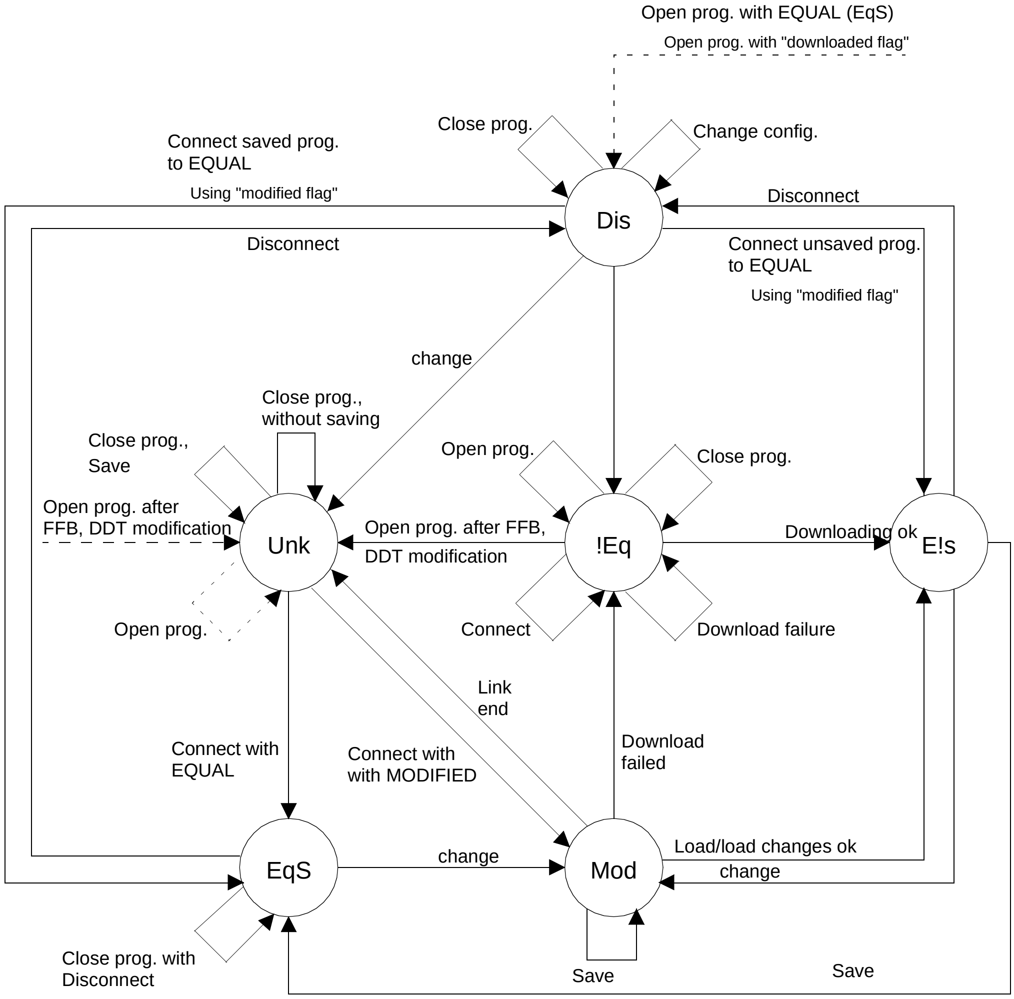
If a project is open and a connection between the programming device and the PLC is to be created, a consistency check is automatically carried out among the program, EFBs and DFBs on the programming device, and the PLC. The result of this check (**EQUAL**, **MODIFIED** or **NOT EQUAL**) is displayed in the status bar and written in a file. This file can be found in the Concept project directory and is designated PROJECTNAME.RMK. It is for internal use only and automatically changes its content. The meaning of the individual entries can be found in the following diagram.

Status descriptions

Status descriptions:
- EQUAL
  The program on the programming device and the PLC is consistent.
- NOT EQUAL
  The program on the programming device and the PLC is not consistent. To establish consistency use the menu command **Online** → **Download...** is necessary.
- MODIFIED
  The program on the programming device was modified. The modifications can be made online in the PLC with the menu command **Online** → **Download changes**.
  Note: Even when changes that are not relevant to the code (e.g. creating/ changing comments in IL/ST, moving objects (without affecting logic) exist in FDB/LD/SFC), the **MODIFIED** status is displayed temporarily. When the section is next analyzed (**Project** → **Analyze project**, **Project** → **Analyze section** or **Online** → **Download changes**), the program automatically reverts to the **EQUAL** status (if no changes have been carried out that are relevant to the code). Even if changes that are relevant to the code have been carried out, only these sections appear in the **Download changes** dialog box.

Relationships between states

The diagram shows the relationships between the different program states:



Unk UNKNOWN

Dis DISCONNECTED

!Eq NOT EQUAL

Mod MODIFIED

E!S EQUAL but not saved

EqS EQUAL and saved

Presettings for ONLINE operation

At a Glance

The dialog box **Link PLC to** can be used to specify settings for both the PLC link and ONLINE mode resulting from it.

Access

It is possible to specify which functions will be executed in the ONLINE operation, i.e. which menu commands are available in the **Online** main menu.

Protocol types

To link the programming device and PLC, it is important to know which network the communicating nodes are in so that the correct protocol type is selected.
Use the table to decide which protocol type fits the network link used:

Linking the network nodes	Protocol type
Serial Interface	Modbus
SA85-/PCI85-Adapter	Modbus Plus
NOE-module (on Ethernet-Bus SINEC H1)	TCP/IP
TCP/IP Interface map (32-Bit Simulation)	IEC Simulator (32-Bit)

Note: The programming device can always only be linked to one PLC. This means that before a link is made to another PLC, any existing link must be terminated with the **Terminate Link** menu command.

Modbus Network Link

Introduction

For a Modbus network link, the settings of the modbus interface must correspond with the settings on the PLC.
The interface is edited in the **Modbus Port Settings** dialog (**PLC Configuration** → **Modbus Port Settings...**).

Protocol Settings for Modbus

When the Modbus protocol type is selected, specify further data in the **Protocol Settings: Modbus** range. Specify the Node Address (Node No.) on the PLC and enter this in the corresponding text box. You can determine the transfer mode for communication between the PLC and the host computer.
The following modes are available according to the application:

Application	Mode
Communication with various host devices. The ASCII mode works with 7 data bits.	ASCII
Communication with an IBM compatible personal computer. The RTU mode works with 8 data bits.	RTU

After the serial interface for the Modbus network link has been specified, using the **Settings...** command button, open the **Settings for COMx** dialog. Enter the settings for the interface here, as in the **Modbus port settings** dialog.
Use the **OK** command button to create the ONLINE link.

Modbus Plus Network

Introduction

For a Modbus Plus network connection, enter in the **Protocol settings: Modbus Plus** range whether the 16-Bit IEC-Simulator (**Port 0**) or the Modbus Plus interface (**Port 1**) is being used.
All nodes on the local network are displayed in the list. Additionally, the routing path of the token rotation sequence in the network, which can contain up to 5 Node addresses is displayed. Up to 64 nodes can be addressed on one network, i.e. a routing path address can be between 1 and 64. Several networks can be linked via a bridge.

> **Note:** The node list of a different network can be displayed by double-clicking on a listed bridge.

To pass the program execution to the Modbus Plus device driver, Concept triggers a MS DOS Software Interrupt. The preset Interrupt number for this is 5C (hex).

> **Note:** If no virtual Modbus Plus driver is installed, the virtual MS DOS environment in Windows NT has problems when reacting to the software interrupt. If a share violation (Exception) occurs under certain conditions, change the Interrupt number to 5D (hex) in the MODICON.INI file:

```
[PORTS]
mbp0=5d
```

If the Interrupt 5D from the NTVDM.EXE is activated the share violation should no longer occur.

IEC Simulator (16 Bit)

The simulator simulates a coupled PLC via Modbus Plus. The address of the programming device is displayed in the list in the routing path.
The simulator is active if in the **Protocol settings: Modbus Plus:** area, the option **Port 0** is selected.

> **Note:** When the simulator is active, no further nodes can be displayed.

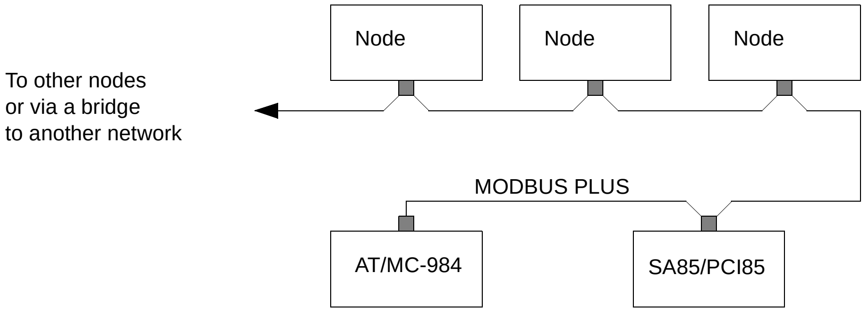
The simulator is only available for the IEC languages (FBD, SFC, LD, IL and ST).

PLC as Modbus Plus Node

When the PLC is a Modbus Plus node, the address which the PLC has in the routing path is displayed in the list. This address corresponds to the node address which is set with a rotary switch on the back of the CPU.

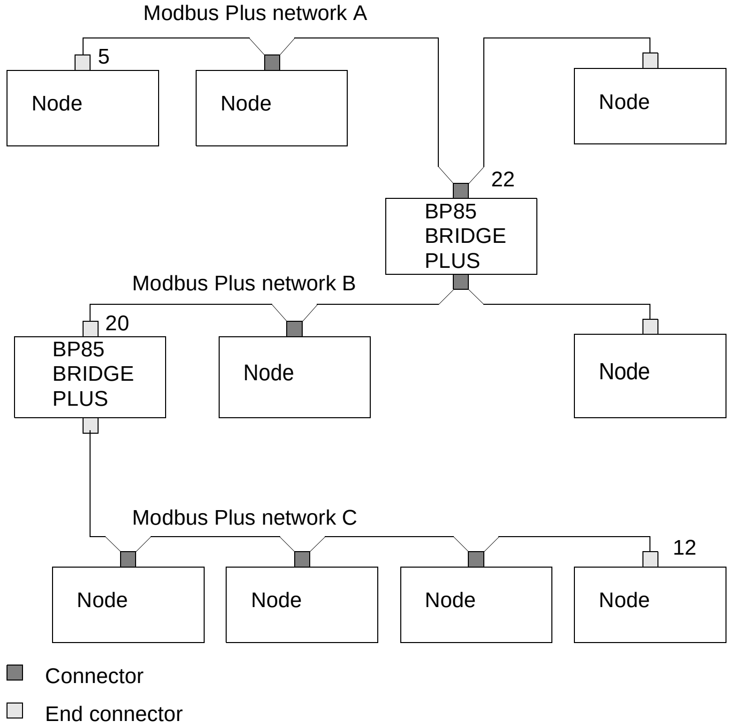
SA85/PCI85 as Modbus Plus Node

The SA85 module is a Modbus Plus adapter for an IBM-AT or compatible computer. The port address is displayed in the list. The address shows in which network the SA85/PCI85 is installed.
Displaying a routing path with SA85/PCI85:



Bridge Plus as Modbus Plus Node

A Bridge Plus (BP85) links nodes within two Modbus Plus networks. The Bridge is displayed in the list box, and the next Modbus Plus network can be accessed by double-clicking on the Bridge.
Displaying a routing path with a Bridge Plus BP85:



Example:
The example shows a routing path across 3 Modbus Plus networks. The task is to send a message from node 5 in network A to node 12 in network C.
The routing path here is 22.20.12.00.00 and it is made up as follows:

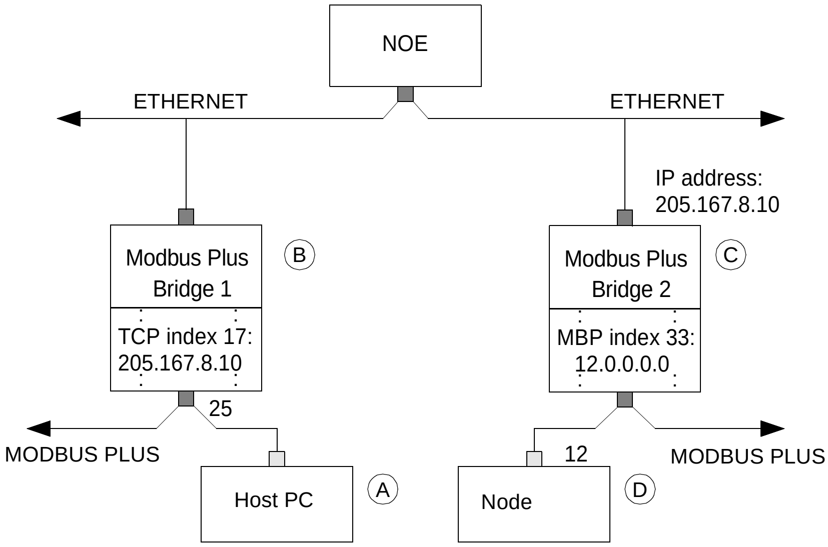
Path	Meaning
22	The first address contains the network A Bridge Plus address from Network A from output node 5. This means the message is sent from output node 5 across this Bridge to the next network, B.
20	The second address contains the Bridge Plus address of the next network, B. Here, the message is sent from network B to the third network, C.
12	The third address contains the address of node 12, the destination segment.
00.00	Addresses four and five are set to 0 because there are no further forwarding addresses.

Bridge as Modbus Plus Node

A link between the Ethernet and the Modbus Plus network or between two Modbus Plus networks is created via the Modbus Plus Bridge.
The Modbus Plus Bridge should be regarded as a host computer and must be configured in the **Protocol settings: TCP/IP** area. Enter the IP address or the host name of the Bridge, and finally in the text box **Protocol type:** change to Modbus Plus network setting.
The Modbus Plus Bridge is only listed in the list of nodes in the Modbus Plus network as a host name which was previously entered in the **Protocol Settings: TCP/IP** area. A double-click on the corresponding host name opens the **Modbus Plus Bridge** dialog box for 5 byte routing path configuration.
Further action in the dialog box can be found in the chapter"*Modbus Plus Bridge, p. 582*".

Example:

In the dialog box **Modbus Plus Bridge (See *Modbus Plus Bridge, p. 582*)**, create the routing path 25.8.17.33.0, which defines the following link (from A to D):



Modbus Plus routing path:
25.8.17.33.0

A The message sent from the host computer contains the 5 byte Modbus Plus Routing Path. The first byte with the node address of the host computer refers to the Modbus Plus Bridge linked to it. The Modbus Plus Bridge 1 receives the message on internal path 8, as specified in byte 2.

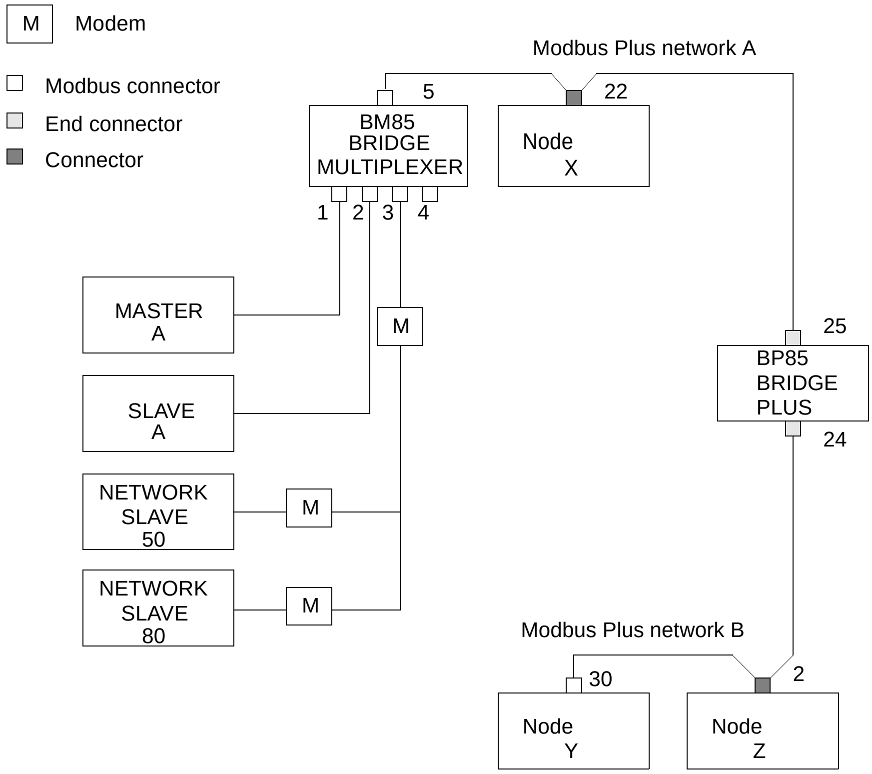
B The TCP Index No. 17 (byte 3) administered in the Modbus Plus Bridge passes the message on to the configured node with the IP address 205.167.8.10. In this case the node with this IP address is another Modbus Plus Bridge.

C Modbus Plus Bridge 2 receives the message. The MBP Index No. 3 given in 4 byte and administered by the bridge passes the message on to the configured Modbus Plus node. In this case the node 12.0.0.0.0.

D The message has reached its destination, Modbus Plus node 12.

Bridge Multiplexer as Modbus Plus Node

The BM85 Bridge Multiplexer links up to four Modbus devices or Modbus networks to a Modbus Plus network.
See also "User's Guide BM85 Modbus Plus Bridge/Multiplexer."
Displaying a routing path with a Bridge Multiplexer BM85:



Modbus Plus Bridge

At a Glance

Enter the 5 byte routing path which defines the link from the host computer to the Ethernet node in this dialog box.

Making settings

The following table describes how to define the routing path:

Setting zone	Routing path byte	Meaning
Bridge Path	2. Byte	A maximum of 8 links can go out from the Bridge to the other network, and one of these should be selected.
IP routing byte	3. Byte	Enter an index no. which is assigned to an IP address. This IP address should correspond to an Ethernet node address where the message is then sent. If this IP address is being sent to another Modbus Plus Bridge in the Ethernet,another node address (MB+ routing byte) must be given for it to be transferred further into the Modbus Plus network.
MB+ routing byte	4. Byte	If a link is displayed between two Modbus Plus networks via two Modbus Plus bridges, the index no. of the Modbus Plus node must be entered here. This index no. is also assigned to a node address. If there is no link across a different bridge, the value "0" is entered.
Complete address	5. Byte	The whole 5 byte routing path is displayed according to the setting. The first byte is then automatically adjusted to the node address of the host computer.

Modbus Plus index no.

The assignments of the Modbus Plus index no. are pre-set and can be selected between 0 and 255. Note that index no. 255 is reserved for specific operations. When this index no. is selected, data selection or loading is permitted between a TCP/IP node and the Modbus Plus Bridge via an internal command. Index nos. 250 – 253 are reserved and cannot be used.
The index in the Modbus Plus routing path is shown in the following table:

Index	Modbus Plus routing path
1 ... 64	1.0.0.0.0 ... 64.0.0.0.0
65 ... 128	2.1.0.0.0 ... 2.64.0.0.0
129 ... 192	3.1.0.0.0 ... 3.64.0.0.0
193 ... 249	3.2.1.0.0 ... 3.2.57.0.0

TCP/IP Index No.

The assignments of the TCP index no. follow automatically after the IP address of the Modbus Plus Bridge has been specified in the **Link** → **Protocol Settings: TCP/IP** dialog box. Each index is assigned to an IP address where the first 3 bytes are assigned to the first 3 bytes of the Modbus Plus Bridge IP address. The 4th byte is counted upwards from 1 to 255 at the most.
Example:
For a Modbus Plus Bridge IP address of 205.167.4.65, the TCP/IP addresses are automatically pre-set, as in the following table:

Index	IP address
1	205.167.4.1
2	205.167.4.2
...	...
255	205.167.4.255

Note: Refer to the "174 CEV 200 30 TSX Momentum Modbus Plus to Ethernet Bridge User Guide" for a detailed description of the Ethernet Bridge.

TCP/IP-Network Link

Introduction

For an Ethernet link, select the protocol type TCP/IP in the **Connect to PLC** dialog box.

Protocol Settings for TCP/IP

To connect to other Ethernet nodes, specify the IP address or the host name of the Ethernet node in the **Protocol Settings: TCP/IP** range.
To connect to the Ethernet via Modbus Plus node, specify the IP address or the host name of the Modbus Plus Bridge in the **Protocol Settings: TCP/IP** range (see also "Bridge as Modbus Plus Node (See *Bridge as Modbus Plus Node, p. 580*)").

Connecting Quantum to the Ethernet

You can connect the Quantum to the Ethernet Bus by configuring the NOE module. By doing this, it is possible to communicate with other automation components in the Ethernet Bus system via the host computer.

Connecting IEC Simulator (32 bit)

Introduction

The simulator simulates a PLC connected via TCP/IP, where the signal status of the I/O modules can also be simulated. Up to 5 host computers are connected to the simulated PLC at the same time.
To activate the simulator, select the protocol type IEC simulator (32 bit) in the **Connect to PLC** dialog box.

Protocol Settings for IEC Simulator (32 bit)

The simulator is active, if you specify the address of your TCP/IP interface board in the **Protocol Settings: IEC Simulator (32 bit)** range.
The TCP/IP address can be obtained on the title bar of the Concept simulator program PLCSIM32.

Note: At the present time the simulator is only available for IEC languages (FBD, SFC, LD, IL and ST).

State of the PLC

At a Glance

With a network link, the state of the PLC is displayed in the list of nodes in the Modbus Plus network in the **Connect to PLC** dialog box.

States of the PLC

All the states which can arise are listed in the following table:

State	Meaning
Running	Identifies a PLC with a program running.
Stopped	Identifies a PLC with a program which has stopped.
Unknown	Identifies an unknown PLC.
Not configured	Identifies a PLC without a hardware configuration, i.e. no online functions are possible.

20.3 Setting up and controlling the PLC

At a Glance

Overview

This chapter contains information about setting up and controlling the PLC.

What's in this Section?

This section contains the following topics:

General Information

Introduction

The PLC and CPU functions can be controlled in ONLINE mode. The PLC must be connected to the host computer to establish ONLINE mode.
You can control the PLC directly with the following commands:
- Set Scan Time
- Single Scan Function
- Clear Controller
- Set Clock
- Run Optimized Solve
- Save in Flash
- Set Password for PLC

The commands for setting up and controlling the PLC can be found in **Online** → **Online Control Panel**.

Setting the Time for Constant Scans

Introduction

A constant cycle time for processing the user program can be specified in the **Online** → **Online control panel** → **Invoke constant sweep** → **Constant Sweep Settings** dialog box.
However, if the actual cycle time is longer than the constant cycle time specified the system ignores the user settings and uses the normal cycle running time (**Cycle time in free running mode**).
If a constant cycle time is selected which is longer than the actual cycle time, the control will wait during each cycle until the set cycle time has been reached.

Note: Inputs/outputs connected via communication experts may not be used for updating constant I/O requests, as there can be highly variable I/O response times.

Note: This function cannot be performed when there is a link with the simulator.

Selection condition

This dialog box is only available if the link has been established between the PLC and the programming device (ONLINE mode).

Settings for constant cycle

A tab (4x) must be specified first to determine the constant cycle. You also need to enter the scan time (10-200m) that is allocated to the register.

Note: The scan time increases if several windows are open in Concept, e.g. several sections are displayed in animation mode. Therefore if you are using several windows you should reduce the scan time.

Exiting Constant Scan

After starting the constant scan with the **Invoke constant sweep...** changes the designation of the command button in **Cancel constant sweep...**. Clicking on this command button exits the function.

Single Sweeps

Introduction

You can specify single sweeps times for processing the user program in the**Online** → **Online Control Panel** → **Single Sweep On…** → **Settings for Single Sweeps** dialog box.
After the specified number of scans has been performed the logic editing stops. This function is helpful for diagnostic purposes. It allows the checking of edited logic, modified data and calculations that have been carried out.

WARNING

It can lead to unsafe, dangerous and destructive operations of the tools or processes that are attached to the controller.

Single sweeps should not be used for searching for errors in controlling machine tools, processes or material maintenance systems if these are running. When the number of scan times given has been processed, all the outputs will be retained in their last state. Since no more logic editing is taking place, the controller ignores all input information. Therefore the single sweeps function should only be used for searching for errors during start up.

Failure to follow this precaution can result in death, serious injury, or equipment damage.

Selection Condition

This dialog box is only available if the link has been established between the PLC and the host computer (ONLINE mode). Single sweeps are only performed in PLC RUN mode.

Settings for Single Sweeps

To determine the single sweeps, the scan time (10 – 200ms) and the number of scans being performed must be specified. A maximum of 15 single sweeps is possible.

Execution of Single Sweeps

After the scan time and number have been specified you can perform the single sweeps with the **Trigger Sweep** command button.

Note: The **Trigger Sweep** command button is only available in PLC RUN mode.

Exiting Single Sweeps Function

After the single sweeps function has been started with the **Single Sweep On** command button, the designation of the command button changes to **Single Sweeps Off**. Clicking on this command button terminates the function again, and the **Settings...** and **Trigger Sweep** command buttons no longer appear in the dialog box.

Deleting memory zones from the PLC

At a Glance

Specific memory zones can be deleted from the PLC by activating the corresponding options key in the**Online** → **Online Control** → **Delete PLC...** → **Delete PLC contents** dialog box.
The menu command **Load...** can be used to load the deleted memory areas back onto the PLC.

Condition for dial-in

This dialog box is only available if the link has been established between the PLC and the programming device (ONLINE mode) and the PLC is in STOP mode.

Deleting a configuration

If the hardware function of a PLC is deleted, no further online functions can be performed. The NOT CONFIGURED and NOT EQUAL TO modes are displayed in the status bar.

Deleting a program

If the user program is deleted in the PLC, the PLC cannot be started. The NOT EQUAL TO state is displayed in the status bar.

Deleting state RAM

If the state RAM is deleted, the initial values of the located variables in the PLC are set to 0.

Speed optimized LL984-Processing

At a Glance

A speed optimized LL984 Processing can be optimized in the dialog box**Online** → **Online Control** with the **Opt. processing in** command button.
After the command button is activated its designation changes in **Opt. Processing out**. This means that a click on this command button will deactivate the speed optimization which is running again.

Note: This function only influences the LL984- program.

Condition for dial-in

This dialog box is only available if the link has been established between the PLC and the programming device (ONLINE mode) and the PLC is in STOP mode.

Save To Flash

Introduction

For data protection purposes, you can save parts of the RAM in the PLC's Flash-EPROM. After a power failure, the contents of the Flash-EPROM is loaded back onto the CPU RAM for the restart.

WARNING

Modified process status after next start!

It is important to choose the right time for saving to Flash, as there could be signal values in the Flash memory which are downloaded later following a power failure, and which do not correspond to the process status for the next start.

Failure to follow this precaution can result in death, serious injury, or equipment damage.

Selection condition

This function is available when using all 140 CPU 434 12 and 140 CPU 534 14 TSX Compact, Momentum and Quantum modules.
This function is not available with IEC Hot Standby operation with Quantum.
The Flash memory function is not available when using the simulator.

Procedure

Carry out the following steps to Save To Flash:

Step	Action
1	In the **Flash Type** area, select the **Internal** or **PC Card** option button depending on the hardware used. **Note:** Applications that require more than 480 kBytes must be saved in the PC Card Flash.
2	In the **Controller State** area, select the operating mode (RUNNING or STOPPED) the PLC should be in after a restart.
3	Check the **Allow Editing After Power Up** check box if you want to edit the uploaded Flash program when the voltage supply returns. **Caution:** Since these later modifications were not downloaded onto the Flash-EPROM, this data is lost if there is a power failure.
4	Check the **Save State Ram** check box if you want to save all 4x registers to Flash-EPROM. **Note:** This selection is not available with the Momentum family, i.e. all applications are always downloaded to Flash-EPROM.
5	If you have checked the **Save State Ram** check box, you must enter the number of registers to save in the **Number of registers to save** text box. The corresponding register area, which is downloaded onto Flash-EPROM, is then set from the 400001 address.
6	Select the **Save To Flash** command button to load the user program, the configuration, the IEC programming initial values from the RAM to the Flash EPROM.

Edit Flash program

As soon as the **Allow Editing After Power Up** check box is checked, on saving to Flash, information is loaded to the Flash-EPROM, which after uploading the contents of Flash (e.g. in the case of the return of the power supply) allows the program to be edited. Since these later modifications were not downloaded onto the Flash-EPROM, this data is lost if there is a power failure. To prevent this, changes should be downloaded to Flash-EPROM by using the **Save To Flash** command button.

Modification of the Flash program is not allowed.

As soon as the **Allow Editing After Power Up** check box is unchecked, the program can be modified after reading the Flash contents (e.g. in the case of the return of the power supply), but cannot be loaded to the Flash EPROM.
Modifying the program leads to the following reactions when uploading:

Procedure:	Changes protected with Download changes...	Changes protected with Save project	The following status is established after connection:
a)	Yes	No	EQUAL
b)	Yes	Yes	NOT EQUAL

If the EQUAL status is established in the above case a), the contents of the host computer are different from the contents of the Flash-EPROM. After a power failure the Flash-EPROM is uploaded, resulting in the loss of all changes.
If the NOT EQUAL status is established in the above case b), the contents of the Flash-EPROM are different from the contents of the host computer. After a power failure the Flash-EPROM is uploaded, resulting in the loss of all changes.

Note: To download a program change to Flash EPROM again, the **Save To Flash** command button must be available again. Specific steps must be carried out to do this, as described in the *Reactivate flash save, p. 594* section.

M1 Ethernet CPU

The password protected application is automatically downloaded on each switching on/off cycle. This procedure cannot be undone if you forget the password. The PLC must be sent for repair.

Reactivate flash save

Introduction

If you have not checked the check box in Flash Save **Allow Editing after Power Up** the program saved in Flash EPROM can no longer be changed. After a power failure the Flash-EPROM will finish on restarting the PLC. However, the command buttons **Save to Flash** and **Clear Flash** are not available.

Reactivate Flash Save

In order to enable the Flash Save again, the following steps are necessary:

Step	Action
1	Turn off the controller.
2	Compact CPUs: Set the "Memory Protect" switch (Memory Protect) to ON. Quantum CPUs: Set the switch to the "stop" position.
3	Turn the controller on again.
4	Compact CPUs: Set the "Memory Protect" switch (Memory Protect) to OFF. Quantum CPUs: Set the switch to the "start" position.
5	Make the link between the host computer and the controller (**Online** → **Connect...**).
6	Open the dialog box Save to Flash (**Online** → **Online control panel** → **Flash Program...**). **Result:** The command buttons **Save to Flash** and **Clear Flash** are now available again.

Set PLC Password

Introduction

You can use a password to prevent the PLC being written to without permission. Before you can set a new password, however, you must first download the configuration to the PLC. Then enter the password that is to loaded to the PLC. The password is now saved so that password protection operates when a connection is made between the host computer and the PLC (password required).

> **Note:** When setting a Quantum password, a specific time can also be set for the automatic cancel function in the **Quantum Security Parameter** dialog box. This function is found in the preference setting **Never**. This function means that the user is logged out after the time specified as soon as no read or write access occurs from the programming device to the PLC through this connection within the predefined amount of time.

Valid characters for the PLC password and user name

The following characters are permitted within the character length of 616 characters:
- a ... z
- A ... Z
- 0 ... 9
- _

> **Note:** Spaces, umlauts (e.g.: ä, ö, ü) and special characters are not allowed!

Selection conditions

This function is available when using all TSX Compact CPUs, a Quantum CPU 434 12 A/534 14 A or a Momentum Ethernet CPU.

Note

The following passwords can be assigned in Concept:
- PLC password
- Concept Password (See *Changing Passwords, p. 708*) (in Concept Security)

Set new PLC password

To set a new PLC password, proceed as follows:

Step	Action
1	Using **Online** → **Download** load the configuration onto the PLC
2	Using **Online** → **Online Control Panel...** → **Set PLC password...** open the dialog **Change PLC Password**.
3	Enter your new password in the **Enter New Password:** text box.
4	Enter the new password in the **Confirm New Password:** text box again.
5	Enter the user name in the **User name** text box, e.g. "anyname".
6	Press the **OK** command button. **Reaction:** The dialog box is closed and the password is automatically downloaded to the PLC

Change Old PLC Password

To change an old PLC password, proceed as follows:

Step	Action
1	Using **Online** → **Online Control Panel** → **Set PLC password...** open the dialog **Change PLC Password**.
2	Enter your old password in the **Enter Old Password:** text box.
3	Enter your new password in the **Enter New Password:** text box.
4	Enter the new password in the **Confirm Password:** text box again.
5	Enter the user name in the **User name** text box.
6	Press the **OK** command button. **Reaction:** The dialog box is closed.
7	Using **Online** → **Download** load the configuration onto the PLC **Reaction:** The password was loaded onto the PLC, and will be requested the next time the PLC and the host computer are connected.

If You Forget Your Password

If the PLC password has been forgotten the procedure depends on the PLC platform used.

Quantum and Compact:

Step	Action
1	Switch off the power supply to the PLC.
2	Move the Memory Protect switch on your hardware module to the MEM_PROT position.
3	Remove the lithium battery from the PLC.
4	Wait 5 minutes and then switch on the power supply to the PLC again. **Reaction:** By doing this, the battery backup RAM is deleted without downloading the PLC program from Flash-EPROM. In this way, the start status of the PLC (configuration-free and without log on password) is re-established.
5	Continue with the step table *Set new PLC password, p. 596*.

Momentum without Flash:

Step	Action
1	Switch off the power supply to the PLC.
2	Remove the battery from the interface adapter.
3	Wait 5 minutes and then switch on the power supply to the PLC again.
4	Continue with the step table *Set new PLC password, p. 596*.

Momentum with Flash:

Step	Action
1	Switch off the power supply to the PLC.
2	Send the module back to the product manufacturer (Schneider Automation GmbH).

20.4 Selecting Process information (status and memory)

At a Glance

Overview

This chapter contains information about selecting the process information.

What's in this Section?

This section contains the following topics:

General information

At a Glance

Certain processes and their storage occupancy can be controlled during operation of the automation equipment.

Note: Errors can occur when selecting a configuration that has been generated by another configuration tool (e.g. SyCon, CMD). The selection is based on removing the memory, whereby this is not always compatible with the other software programs. Therefore please use the Modsoft Converter to transfer the Modsoft application according to Concept.

Read status bits.

Status bits provide information about the hardware communication with other modules as well as existing errors in the running of the program. The user specifies the status register already during configuration. In this register , status bits that change their state as soon as a faulty signal is set in the process or a timeout word is not observed are saved. The user can recognize via defined status states (0 or 1) whether the process is faulty.

Read storage occupancy

The user can control the storage occupancy for the current project in the memory statistics. In an overview the total memory, free memory space and used memory for the user program, as well as the user files and FFB libraries are displayed.

PLC state

At a Glance

All the PLC states are displayed in the multi-page dialog box.
There are 67 pages altogether, containing various state information

Condition for dial-in

This function is only available if a link has been established between the PLC and the programming device. When the simulator is active the PLC states cannot be retrieved.

Programming states

The following status information is obtained through the programming:
- Number of segments
- End of logic pointer address
- Run/Download/Debug Status

Hardware states

The following state information is given about the hardware:
- CPU state
- S911 Hot Standby State
- Machine State
- State of the I/O processor
- Quantum I/O state
- DIO-State

Error codes

The following state information is given about errors arising:
- Machine stop code
- Quantum start error code S908

Transfer and communication states

The following state information is given about transfer and communication executions:
- Data transfer state
- Message transfer state
- Communication state

Cable A + B states

The following state information is given about the A + B cable:
- Cable A + B error counter
- A + B global state
- Cable A + B communication error counter

Memory Statistics

Introduction

An overview of the memory data for the open project is given in the **Memory statistics** dialog box. The current scan time is also displayed if a real PLC is used (and not the simulator).

Total IEC memory

The memory statistics cover the following information:

Total IEC memory	Meaning
Configured	The displayed value corresponds to the value specified in the **PLC Selection** dialog. **Note:** If you use a simulator, the total memory is not given.
Used	The displayed value corresponds to that of the IEC program memory space used by the user program. Note: If you use a simulator, the used memory space is not given.

Modifying Total IEC Memory Size

The total IEC memory consists of the IEC program memory and the global data. Additional space is required in the total IEC memory for program extensions and for the administration of program modifications. The general recommendation is to set the value so that 20-30% of the value entered in the **Used** text box also remains free.

Note: Changes can only be made offline and are only accepted once the program has been downloaded to the PLC.

IEC Program Memory

The values displayed correspond to the memory space used for
- Program code
- EFB code
- Program data (section and DFB instance data)

Global Data

The memory statistics cover the following information:

Memory space	Meaning
Configured	The displayed value corresponds to the value specified in the memory space for unlocated variables in the **PLC Selection** dialog.
Used	The displayed value corresponds to the value from the memory space for the declared unlocated variables used by the user program.

Changing the Memory Size for Global Data

You can change the memory size of the global data. It should be noted that an increase in the global data size decreases the IEC program memory size. Each object, e.g. FFB instance, variable, step etc., takes up several bytes in the IEC program memory.
Because more memory space is not automatically gained by deleting unlocated variables, it is recommended that sufficient memory space is planned. The general recommendation is to set the value so that 20-30% of the value entered in the **Used** text box also remains free.

Note: Changes can only be made offline and are only accepted once the program has been downloaded to the PLC.

Scan Time

The value displayed corresponds to the current scan time. With the first call, the I/O station is standardized so that a scan time of 0 ms/scan is specified. After initialization, the scan time is calculated as an average value.

Note: If you are using the simulator, the scan time is not given. The display **na** means "not available".

20.5 Loading a project

At a Glance

Overview

This chapter contains information about loading a project.

What's in this Section?

This section contains the following topics:

General information

At a Glance

To carry out an online command a transfer has to be made to the PLC after setting up or changing sections. Otherwise a complete project can be transferred from the PLC to the programming device. As soon as the user program is consistent on the programming device and the PLC, the EQUALS status is displayed in the status bar. The status display MODIFIED identifies the program in which at least one section has been changed or where changes to the variable editor were performed. With command button **Download changes...** the consistency between the programming device and the PLC is restored. Status display NOT EQUALS identifies a program in which critical changes were performed. Critical changes are for example changes to EFBs, DFBs or derived data types. With command button **Download...** the consistency between the programming device and the PLC is restored.
Loading, loading changes and selecting are not possible in the animation mode.
With command button **Select...** the following project areas can be selected from the PLC:
- Configuration
- IEC sections
- 984 Ladder Logic sections
- ASCII messages
- State RAM
- Initial values
- Extended memory

Process for loading

Loading the PLC can take place in two parts:
1. The exportable code (machine code) is always loaded onto the PLC.
2. The complete compressed user program is loaded onto the PLC

Note: The user program, consisting of user defined EFBs, DFBs, derived data types and the program (variables, sections, etc.), is only loaded onto the PLC if in dialog **Options for generating codes** (**Project** → **Options for generating codes...**) check box **Contain IEC selection information** was activated beforehand. The option to also load the comments contained in the check box onto the PLC, thereby making them available as selection information, is available, as well.
During selection the entire user program can be transferred from the empty project to the programming device.

Loading

At a Glance

With command button **Download...** The configuration, the entire user program (IEC or LL984 section) ASCII messages (only with Concept for Quantum) and the state RAM with the initial values of a project can be sent in the PLC. This establishes consistency between the user program on the programming device and the PLC so the online functions are executable.

Loading single parts onto the PLC

Single parts to be loaded onto the PLC can be selected.
The following table contains the available options and their meaning:

Option to be loaded	Meaning
Configuration	This option sends the hardware configuration to the PLC. **Note:** The Hardware Configuration can only be sent to the PLC when a corresponding access privilegehas been authorized. This option is not available with a Modbus Plus connection.
IEC Sections	This option sends the code from all the sections created with an IEC programming language (FBD, SFC, LD, IL, ST) to the PLC.
984 Ladder Logic	This option sends the code from all the sections created with an LL984 programming language to the PLC.
ASCII messages	This option sends ASCII messages for Ladder Logic to the PLC. **Note:** This function is only available when using Concept for Quantum.
State RAM	This option sends the State RAM to the PLC.
Only initial values	This option sends only initial values from the user program to the PLC. The initial values can only be loaded together with the State RAM. This means that the check box is only available when loading the State RAM.
Extended memory	This option assigns the PLC an extended memory allocation (6x-Referenzen) zu. **Note:** This function is only available when using Concept for Quantum.

Loading IEC selection information

To receive a complete project when selecting from the PLC **Options for code generation** the check box **IEC selection information** must be activated in the dialog box before loading. If this check box is not activated only the executable code (machine code) is loaded onto the PLC.

If loading is not possible…

There are several possibilities why loading is not possible:
- An active screen saver can lead to loading errors. It is therefore recommended to deactivate the screen saver.
- If loading the problem is not possible due to insufficient program data memory, the memory size can be optimized.*Main structure of PLC Memory and optimization of memory, p. 113*.

> **Note:** If, while loading the program, a warning appears due to inconsistent DFB versions, use the menu command **Project** → **Synchronize nested DFB versions**.

Download Changes

Introduction

Download changes is always used if sections have been changed, added or deleted, whether online or offline, and the program is in MODIFIED mode. In this way the changes are indicated and can be transferred to the PLC.
The changes are loaded into the PLC and the consistency between the user program on the programming device and the PLC is restored.

Changes, which have no affect on the logic of the program (e.g. renaming a step name, renaming a section, renaming a variable, graphic move of a module etc.) are not loaded into the PLC with the **Download changes** function. If non-logic related changes are to be loaded into the PLC (so that, for example, these changes are available after the PLC has been uploaded to the PC), the entire project must be loaded into the PLC with **Online** → **Download**. Only then are these changes available after PLC uploading.)

If the changes cannot be downloaded because there is too little memory in the PLC, there are 2 possibilities for proceeding:
- Sequential loading of modified sections
- Optimize Project

> **Note:** If a warning appears when the program is being downloaded, due to inconsistent DFB versions, execute the menu command **Project** → **Synchronize versions of nested DFB** .

ID for specific sections

The following sections contain additional ID information as they are different from cyclically set sections:
- **E** for "**E**vent Section" (I/O Event and Timer Event Section = Interrupt-Section)
- **T** for "**T**ransition Section"

Sequential loading of modified/new sections

The user can download changed/new segments onto the PLC one after the other.

When segments are downloaded sequentially, the following points must be noted:
- If the constants value has been changed, it is not possible to download the changed segments sequentially.
- All deleted IEC segments will be automatically deleted the first time the user downloads sequentially onto the PLC.
- All initial values of new variables, all modified values of literals are automatically loaded onto the PLC on the first sequential loading.
- If new sections already contain used variables, the value of these variables remains.
- When closing a project ensure that it is saved before loading changes onto the PLC. Otherwise it might not be possible to continue the project after it is reopened with the remaining changes loaded, or there will be "newer" sections (previously loaded changes) on the PLC than on the programming device.

CAUTION

Danger of unwanted and dangerous process conditions

Loading sections sequentially on a running PLC can lead to unwanted and dangerous process states. It is therefore recommended to stop the PLC during sequential loading.

Failure to follow this precaution can result in injury or equipment damage.

Modified Initial values

Modified initial values are no longer loaded onto the PLC. The initial value, which was transferred to the PLC during the first load (**Download.../Download changes...**), cannot be overwritten by the **Download changes...** menu command. The initial values can however be changed in the Reference data editor.

Procedure for sequential loading

The procedure for downloading changes sequentially is as follows:

Step	Action
1	Stop the PLC with **Online** → **Online control panel** → **Stop controller**.
2	Select the segment(s) which must be downloaded from the list.
3	Confirm using **OK**.
4	Call up the dialog box again and repeat the process until all modified/new sections are loaded onto the PLC and **EQUAL** mode is reached.
5	Start the PLC with **Online** → **Online control panel** → **Start controller**.

Loading IEC upload information

If, in the **Code generation options** dialog box, the check box **Include IEC Upload information** is checked, IEC Upload information is loaded onto the PLC with the **Download changes...** menu command.

Optimize the Project

It may be possible to eliminate existing gaps in the program data memory management of the PLC with the menu command **Optimize project** and enable loading again in this way. However, the PLC must be stopped and the complete program must be downloaded again. Furthermore, it may be necessary to adjust the size of program data memory, see Memory statistics (See *Memory Statistics, p. 601*).
It is still possible to optimize use of the program data memory with the menu command **Online** → **Memory statistics**.

CAUTION

Modifications are only transferred when the program is loaded onto the PLC.

After optimizing the project or modifying the program data memory size the PLC must be stopped and the program loaded onto the PLC.

Failure to follow this precaution can result in injury or equipment damage.

Uploading the PLC

Introduction

With command button **Upload...** the configuration, the entire user program (IEC or LL984 section), the ASCII messages and the state RAM with a project's initial values can be transferred from the PLC to the host computer.

Note: Upload information (PLC configuration), which was generated by the Software programs as Concept, is possibly erroneous. The selection is based on removing the memory, whereby this is not always compatible with the other software programs. Please use the Modsoft converter for transferring your Modsoft application to Concept.

Reading Individual Parts from the PLC

Individual parts to be loaded from the PLC to the host computer can be selected. The following table contains the available options and their meaning:

Option to be loaded	Meaning
Configuration	This option sends the hardware configuration to the host computer. **Note:** The hardware configuration can only be sent from the PLC when a relevant authorization is granted in the Access Rights. This option is not available with a Modbus Plus connection.
IEC sections	This option transfers the revertive presentation information of all the sections created with an IEC programming language (FBD, SFC, LD, IL, ST) to the host computer. In this way, however, no current signal values from variables and registers are loaded.
984 Ladder Logic	This option sends the revertive information from all the sections created with an LL984 programming language to the host computer.
ASCII Messages	This option transfers ASCII messages for Ladder Logic to the host computer. **Note:** This function is only available when using Concept for Quantum.
state RAM	This option transfers the state RAM to the host computer.
Initial values only	This option only transfers initial values from the user program to the host computer. The initial values can only be uploaded together with the state RAM, i.e. the check box is only available when the State RAM is activated to upload.
Extended memory	This option transfers the PLC's available extended memory (6x references) into the configuration. **Note:** This function is only available when using Concept for Quantum.

Upload Procedure

Introduction

If the IEC upload information was being taken into account during loading into the PLC (**Project** → **Code Generation Options** → **Include IEC upload information**), a new project containing the IEC upload information is generated in Concept during upload. In this way, the entire user program and user EFB libraries are always downloaded, i.e. individual sections, EFBs etc, cannot be selected for transfer.

Note: During loading (**Online** → **Download Controller**) of the IEC upload information, additional memory is required so that this function should only be used, when you want to upload the project loaded into the PLC again.

Requirements

In order to carry out a PLC upload, an empty project must first be created.
There are several ways of doing this.

Selection	Action
1	You can create an empty project using the **File** → **New project** menu command. Then execute the **Online** → **Upload...** menu command. **Result:** The **Upload to project** dialog is opened. Here you can determine (e.g. D:\NEW\TESTPRJ.PRJ) where the project will be uploaded to. **Note:** You can select a different directory or even create a new directory so as not to come into conflict with existing projects. The preset project name corresponds to the project name downloaded in the PLC and does not necessarily have to be changed.
2	Using the **File** → **Open...** menu command you can create a new project (e.g. D:\NEW\TESTPRJ.PRJ) Then execute the **Online** → **Upload...** menu command. **Result:** The **Upload Controller** dialog is opened.
3	There is **no** project open and you have established a connection with the PLC using the **Online** → **Connect...** menu command. Then execute the **Online** → **Upload...** menu command. **Result:** The **Upload to project** dialog is opened. Here you can determine (e.g. D:\NEW\TESTPRJ.PRJ) where the project will be uploaded to. **Note:** You can select a different directory or even create a new directory so as not to come into conflict with existing projects. The preset project name corresponds to the project name downloaded in the PLC and does not necessarily have to be changed.

Procedure

To upload loaded IEC information, proceed as follows:

Step	Action
1	Open a new project. **Note:**If, during upload, there is a second project still open, it must be closed. In this case a query appears asking whether the project should be saved before it is closed and all changes are lost.
2	Establish a connection between the PLC and the programming unit (**Online** → **Connect...**).
3	Start the upload procedure (**Online** → **Upload Controller...**). **Result:** A window appears in which you can determine the path for the project that is to be uploaded.

Double Designation

Conflicts with existing names can occur during the upload procedure.
Double designation is prevented for each program sequence as follows:

Program sequence	process
User EFB library	A query appears, which can interrupt uploading. If not, a query appears, asking whether the user EFB library should be overwritten, and whether a backup of the old EFB library should therefore be created.
DTY File (derived data types)	A query appears, which can interrupt uploading. If not, the DTY file of the same name is automatically overwritten. No backup is made of the old file.
DFB library	A query appears, which can interrupt uploading. If not, the DFB file of the same name is automatically overwritten. No backup is made of the old file.

20.6 Section animation

At a Glance

Overview

This chapter describes the basic principles for animating sections. The details can be found in the chapters on individual programming languages.

What's in this Section?

This section contains the following topics:

IEC-Sections animation

At a Glance

IEC sections can be animated, i.e. the program’s current states in the PLC /simulator will be displayed.

Animation is possible with both a running and a stationary PLC. Display data is automatically refreshed when the PLC is running. The static state of the program on the PLC is displayed when the PLC is stationary.

Load and **Load changes** is not possible in animations mode. Should these commands be executed, animation will be stopped automatically.

Requirements for animation

Requirements for animation

- The section to be animated in the programming device and the section loaded onto the PLC must be consitent. Otherwise, establish consistency using **Online** → **Download...** (if mode **UNEQUAL**) or **Online** → **Download changes...** (if mode **MODIFIED**).
  Note: Even when the program mode is **MODIFIED**, the sections that have not been changed can be animated. The mode displayed in the footer refers to the program and not to the currently displayed program.
- To animate, the programming device and the PLC must be online. Otherwise, establish the link using **Online** → **Connect...** .

Active animation display

The active animations mode is indicated:

- by a check mark before the menu command, in the **ANIMATED**box on the status bar,
- by a depressed animations button on the symbol bar and
- by the gray window background.

Animating more than one section

If several sections are animated, an animated section is updated in each cycle. This means that the more animations are active, the "older" the values of the individual animations. Additionally, the animation increases the load on the PLC cycle. For this reason, animations that are no longer necessary should be terminated. This also applies to the animation of many variables or very large derived data types.

Note: When coupling using Modbus Plus, it is recommended that no more than 10 sections should be animated at one time.

Note: When coupling using Modbus, it is recommended that no more than 5 sections should be animated at one time.

Animating a disabled section

If a disabled section is animated, the state **INHIBITED** is displayed in the status bar.

Animation of a transition section

If the animated section is used as a transition section for the sequential control (SFC), and the transition (and therefore also the transition section) is not processed, the status **INHIBITED** appears in the transition section.

Changing a animated section into a symbol

If an animated section is changed into a symbol, the animation with the most current values stops, and then restarts automatically once the section is called.

LL984 Programming Modes

Direct Programming

There are two situations that determine how direct mode ladder editing is applied. The first is where there is no open project and you are connected to a PLC that has a valid program in it. When you select the command **Direct Mode LL Editor** the first program in the first segment is displayed. You can see the direct mode status at the right side of the status bar and the network window is labled **984 LL Direct**.

The second case occurs when you have a project open and you are connected to the PLC (but not **EQUAL**). When you select **Direct Mode LL Editor** in this case a dialog is displayed listing segments and the number of networks contained in each. Click on the segment you want click on **OK** and the network edit window is displayed with a window labeled **984 LL Direct**. If you have an orignal edit window it will remain on the display.

Combination Mode

Combination programming occurs when the programming panel is online. Valid program changes are immediately written to both the controller and the program database simultaneously.

20.7 Online Diagnosis

Diagnostics Viewer

Introduction

Using the diagnostics viewer in Concept (**Online** → **Online Diagnostics...**) it is possible to view the content of the PLC diagnostics error buffer.

Selection Condition

The diagnostics viewer is only available if the PLC is in online mode and the EQUAL status has been created between the PLC and host computer.
The diagnostics viewer only works with the SFC, FBD and LD programming languages and with the diagnostics blocks of the EXTENDED group.

Conditions of the Diagnostics Viewer

To activate diagnostics, a supervision time must first be set for the step (Transition diagnostics) or the diagnostics block (Reaction diagnostics). In addition, in the **Code generation options** dialog (**Project** → **Code generation options...**), the **Include diagnosis information** check box must be checked. As a result memory space is prepared on the PLC (max. 64 diagnostics entries) for the diagnostics error buffer.

Behavior of the error buffer

A maximum of 64 events (errors) and a maximum 20 signals per event are read. If the diagnostics error buffer overflows all further signals (21 onwards) are lost. The next event (error) coming is only entered once an event (error) which has gone has been acknowledged in the error buffer.
A diagnostics error buffer overflow is displayed in the dialog status line.

Note: A maximum of 16 events (errors) can be scheduled within one SFC section. All further errors (17 onwards) are lost. The next event (error) is only entered once a past event (error) has been acknowledged in the error buffer.

Transition Diagnosis

Information on this can be found in the *Transition diagnosis, p. 276*section.

Reaction diagnosis

Information on this can be found in the "Diagnostics Block Library" handbook.

Diagnostics viewer

After analysis, the events (errors) and the analyzed signals are written in the buffer and displayed in the diagnostics viewer in Concept.
The following specific information is contained in transition diagnostics:
- Denotes the transition preventing the active step from being executed to the next step.
- Denotes the TRANS type for transition in a PLC section
- Denotes the active step, which is not executed.
- If this is a transition section in the named transition, the analyzed signals are also listed.

The following specific information is contained in reaction diagnostics:
- Denotes the diagnostics block preventing a reaction from being triggered due to incorrect signals.
- Denotes type ACT, PRE, GRP, LOCK, REA for diagnostics blocks
- Diagnostics block drop number
- The analyzed signals are listed.

20.8 Logging Write Access to the PLC

Logging and Encrypted Logging

Introduction

Logging the write access to the PLC can record the following data among others:
- Section name
- EFB/DFB Instance name, FB type name
- Pin-Name
- [variable name] [literal] [address]
- Old value
- New value
- User name (if the Concept (Login) password is activated in Concept Security)
- Date and Time (see also*Address format in LOG file [Logging], p. 1048*)

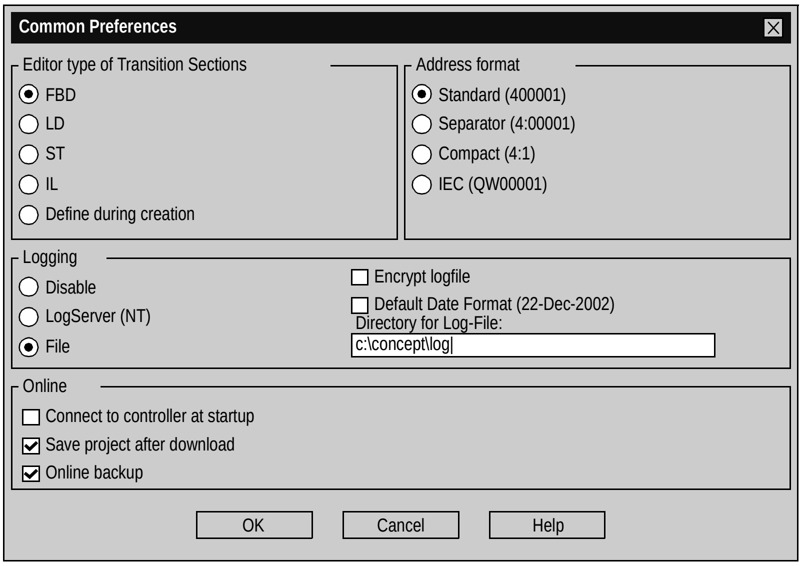
The following logging can be carried out during log-on:
- Modification of the user rights
- Deleted user
- Aborted log-on

Besides the log which can be read in the *.LOG file, an encrypted log can be created in an *.ENC file. The file name is made up of the current date, e.g. 20020723.LOG or 20020723.ENC.
Encrypting the protocol file is done to protect the file contents from being changed. The view tool is only provided so that the user can read the log file. Saving the file in another readable format is not possible. Editing the file so that it is not recognizable is impossible since the ASCII file only displays unrecognizable characters.

Note: Log files are not archived by Concept and no backup files are created.

Log *.LOG

Logging is activated in Concept using the **Options** → **Preferences** → **Common...** → **Common preferences** with the **File** option dialog box. Use the text box **Directory for Log-File:** to define a new path for the log file (e.g. 20020723.LOG). Dialog **Common Preferences**:



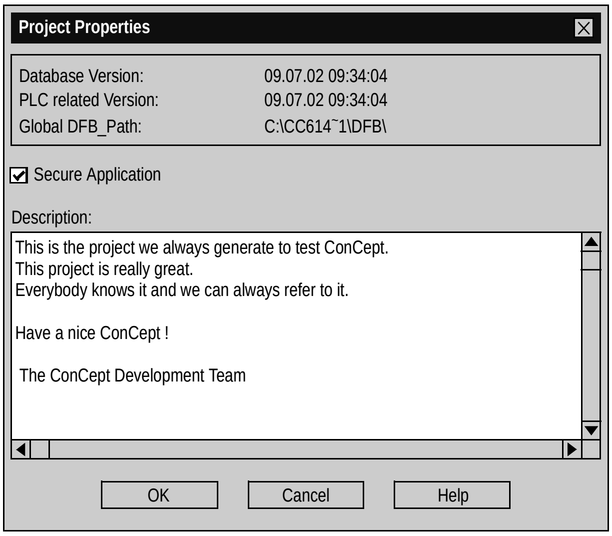
The current logfile can be viewed in Concept with menu command **File** → **View Logfile**.

Encrypted Logfiles *.ENC

In addition, any repetitive strings are displayed in separate encrypted strings in the logfile.
In Concept, the encryption can be activated with two different settings:

- With menu command **Options** → **Preferences** → **Common** → **Common Preferences** and the activation of the check box **Encypt Logfile**.
  Note: The check box is only available if **no** project is open.
- Indirectly with the menu command **Project** → **Project Properties** and the activation of the check box **Secure Application**.
  Note: This dialog box is only available in offline mode.

Dialog **Project Properties**:



If the encryption is activated after an unencrypted logfile (*.LOG) has been created, then a second encrypted logfile (*.ENC) is created. The storage location for the *.ENC file is defined in the **Common Preferences** (**Directory for Log-File:**) dialog box.

Note: Supervisor rights are required for activating the encrypted logging procedure.

View Tool

The View tool can be used for reading encrypted logfiles. Editing and saving so that the file can be read normally is not possible but the logfile can be printed. Supervisor rights are required in this case as well. With menu command **File** → **View Protocol** the View tool is opened automatically if encryption has been activated for the current log.

The logfile is stamped with an electronic signature and the following tests are performed:
- The logfile is created by Concept.
- The logfile is not a counterfeit.

Import/Export

21

At a Glance

Overview

This chapter describes the various import and export options for sections, variables and PLC configurations.

What's in this Chapter?

This chapter contains the following sections:

21.1 General Information about Import/Export

General Information about Import/Export

Export functions

The following export options are available:

Program	Path	Export files
Concept Concept DFB	**File → Export**	• Sections from a source project and import into a target project • Sections from a source DFB and import into a target DFB • Sections from a source DFB and import into a target project • Sections from a source project and import into a target DFB • FBD, SFC and LD sections into IL or ST files • Variable declarations into an ASCII file (Concept only) • PLC configuration (Concept only)
Concept	**Edit → Save as text file...**	• Contents of IL or ST sections into an ASCII file • Definitions of Derived data types from the Data type editor
Concept	**File → Archiving...**	Relevant project files (compressed)
Concept Converter	**File → Export → Configuration**	PLC Configuration

Import functions

The following import options are available:

Program	Path	Import files
Concept Concept DFB	**File → Import**	• Exported sections from a source project or source DFB • Exported or externally created IL/ST files into IL/ST sections • Exported or externally created IL/ST files into FBD/SFC sections (with conversion) • Variable declarations from an ASCII file (Concept only) • PLC configuration exported with Concept (Concept only)
Concept	**Edit → Insert text file...**	• Contents of ASCII files IL or ST sections • Definitions of Derived data types into the Data type editor
Concept	**File → Archiving...**	Relevant project files (decompressed)
Concept Converter	**File → Import**	PLC Configuration

21.2 Exporting sections

Exporting Sections

Introduction

In Concept it is possible to export projects or DFBs selectively from a source project/ source DFB, and if desired, to then import them immediately into the current target project.

Requirements

The project from which the export is to take place must be stable (check using **Project** → **Analyze Program**).

> **Note:** When exporting IL and ST sections, ensure that the settings for nested comments (**Options** → **Preferences** → **IEC Extensions** → **Allow nested comments**) are identical in the source and target projects.

Export range

The following are exported:
- the selected sections with their accompanying variables, DFBs, EFBs and data types.
- In the case of SFC, the accompanying transition sections are also exported.
- The PLC configuration is **not** exported.

Exporting more than one section

When more than one section is exported, a "pseudo SFC" is generated to maintain the execution order. To do this, the following code is generated:

```
INITIAL_STEP   SECTION_SCHEDULER:
    Section1 (N);
    Section2 (N);
       :
    SectionN (N);
END_STEP
```

Exporting FBD, SFC and LD Sections

Using **File** → **Export** → **Program: IEC Text** FBD, SFC and LD sections can be exported to IL and ST. The text languages of both export files follow the grammar of IEC text languages, shown in IEC 1131-3 and in the process tables 52 - 56 of IEC 1131-3.

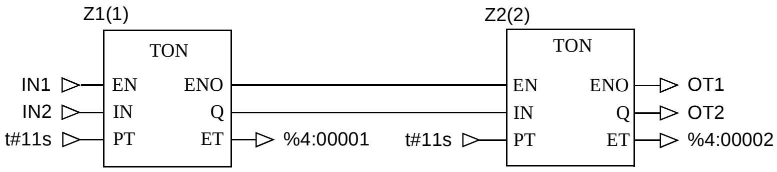
The exported code is displayed in a PROGRAM … END_PROGRAM or FUNCTION_BLOCK ... END_FUNCTION_BLOCK frame and contains all project or DFB variables in a VAR ... END_VAR frame at the start of the file.

If more than one section is being exported, the code separation is expressed as an artificial PLC frame, which is not a component of the original program. It only has one INITIAL_STEP for all sections linked to it as actions (with the identifier N). These actions (sections) are executed every time the step is active, which is always the case. The actions follow as sections, which do not have variable declarations.

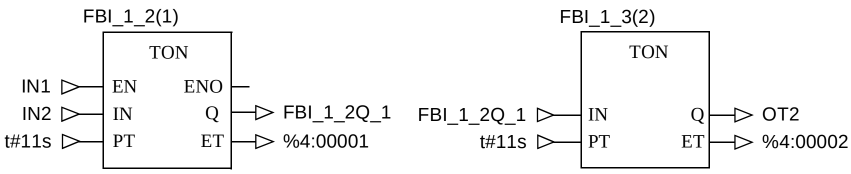
The artificial INITIAL_STEP has the name SECTION_SCHEDULER. It displays the execution order as it was specified in the section execution order dialog box. The artificial SFC frame is left out when re-importing in Concept. The criterion for this omission is the specific name SECTION_SCHEDULER.

The ASCII file can be re-imported into a FBD or SFC section using the IEC text import. Using exports and imports it is possible, for example, to convert a LD section into a FBD section. Importing into a LD section is not possible.

If the EN and ENO optional imports/exports have been used in the FBD/LD sections, they are ignored when exporting to IL/ST.
FBD section logic before export:



FBD section logic after import:



The LD elements "N.C. contact" and "N.O. contact" are converted to AND and ANDNOT.

The ASCII file can, however, also be imported into an IL or ST section using the **Insert Text File** function. In this case, however, manual correction of the files is necessary, since the extensions described above must be removed again from the file.

SFC Export Limitations

The following limitations should be noted when using SFC import:
- Only variables are permitted as actions. Direct addresses cannot be exported.
- Only literals are allowed as time variables for identifiers. Variables are converted to literals with the value 0.
- Transition section names are changed to standard names.
- Step monitoring times and step delay times are lost when exporting.

Exporting IL and ST Sections

Using **Process** → **Save as Text File...** it is possible to export the contents of IL or ST sections into an ASCII file.

This export function is a simple text export function, which can also be performed via the clipboard (cut/copy/paste). Data conversion does not take place. For this reason, the required variable declarations, for example, are not exported with the section contents. If the ASCII files are to be converted to an FBD or SFC section using **File** → **Import** → **Program: IEC Text**, all information necessary for the project (e.g. program frame, section name (see also *Importing (insert file) IL and ST programs into IL or ST sections, p. 651* and *Procedure for "Copying" an IL section from an existing project into a new project., p. 652*)) must be entered manually.

21.3 Exporting variables and derived data types

Exporting variables and Derived Data Types

Exporting variables in "Text delimited" format

Using **File** → **Export** → **Variables: Text delimited** a project's variables can be exported into a ASCII file in text delimited format (refer to *Importing Variables in "Text Delimited" Format, p. 656* and *Importing structured variables, p. 659*).

The ASCII file can be re-imported into a Concept project with the help of the importing text delimited (refer to *Importing Variables in "Text Delimited" Format, p. 656*).

Exporting variables for Factory Link

Using **File** → **Export** → **Variables: Factory Link** a project's variable declarations can be exported into a ASCII file in "Factory Link" format.

If your Factory Link version of Concept is not supported, please call our hotline.

The ASCII file can be re-imported into a Concept project with the help of Factory Link import (See *Importing variables in Factory Link format, p. 662*).

Exporting variables for Modlink

Using **File** → **Export** → **Variables: Modlink** a Modlink configuration file can be generated, which can be used directly in Modlink.

The Modlink configuration file contains all those Located variables, which are selected to be exported in the Variable Editor.

If no Located variables are selected to be exported, an error message appears and a configuration file will not be generated.

Related information about Modlink is found in *Modicon ModLink, User Guide*.

Exporting Derived Data Types

In the data type editor, using **Process** → **Save as text file...** definitions of Derived Data Types can be exported to a ASCII file.

21.4 Section import

At a Glance

Overview

This section describes the import of sections.

What's in this Section?

This section contains the following topics:

Importing Sections

Introduction

In Concept, the possibility exists to export individual sections selectively from a source project or source DFB, and if desired, to then import them immediately into the current target project or DFB:

- Section export from source project, followed by section import into the target project
  This transfers section information, including transition sections at SFC, all used global and local DFBs used, as well as all the variable declarations used.
  Data types defined in data type files are not transferred, (refer to notes).
- Section export from source DFB, followed by section import into the target DFB
  This transfers section information, all global and local DFBs used as well as all declarations of variables, inputs and outputs used.
  Data types defined in data type files are not transferred, (refer to notes).
- Section export from source project, followed by section import into the target DFB
  This transfers section information, all global and local DFBs used as well as all used declarations of unlocated variables.
  Direct address and Located variable declarations must be deleted before export, since they are not authorized in a DFB. Data types defined in data type files are not transferred, (refer to notes).
- Section export from source DFB, followed by section import into the target project
  This transfers section information, all global and local DFBs used as well as all declarations of variables used.
  Declarations of inputs/outputs in this DFB must be deleted before export, since they are not authorized in a Concept project. Data types defined in data type files are not transferred, (refer to notes).

Notes

Attention should be paid to the following notes:

- The imported sections will be inserted at the end of existing sections.
- The PLC configuration is not automatically imported. Instead, it must be imported explicitly (refer to *Import/Export of PLC Configuration using Concept, p. 664*.
- If projects with different local data structures are being imported (different DTY files in the local DFB directories), they must be brought together in an individual DTY file before import. This common file must then be saved in the local DFB directories for source and target projects. Afterwards, open these files to make them known to the individual projects.
- Ensure during import of IL and ST sections that the settings for nested comments (**Options** → **Preferences** → **IEC extensions** → **Nested comments authorised**) are identical in the source and target projects.

Checking the Sections to be Imported

Before the import actually takes place, a check of the following takes place:
- identical project environment (DFBs, EFBs, definition of Derived Data Types)
- existing sections
- existing SFC sections (not authorized in Concept DFB)
- existing step names
- Declarations of inputs/outputs (not authorized in Concept projects)
- Declarations of direct addresses (not authorized in Concept DFB)

If an error is identified, the import is canceled.

Errors that occur subsequently are "irreparable" and will cause the project to close (i.e. all changes made since the last save are lost). Possible errors are:
- Name collisions between variables with different data types
- Name collisions between item names
- other errors

Name collisions between variables with different initial values or direct addresses (located variable) cause a warning. The value of the target project is retained.

Automatic adjustment of standard preset names

An automatic adjustment of standard preset names occurs in the case of:
- Standard generated names, such as SFC step names (S_x_y) and transition section names (TransSection_x_y)
- Standard generated item names (FBI_x_y)
- Position of new DFB inputs/outputs (only with import into Concept DFB)

Specific Changes

During import there are also the following possibilities for performing specific changes, in order to adjust the sections that are to be imported specifically to the target project / target DFB.
- Replacing names (variable names, section names, item names, names in text languages, comments, etc)
- Address offset for located variables and direct addresses in graphic languages (e.g. %3:10 -> %3:20) and text languages (%QW10 -> %QW20)

The following points are excluded from the replace function:
- DFB names
- Index of ARRAYs (e.g. a[1])
- Elements from multi-element variables (e.g. a.dummy)
- In the case of EFBs, the replace function is only used for non-automatically generated names (i.e. Instance names)

Syntax for replacing names and address offset (address shift)

The following syntax applies when replacing names:
- Only entire names will be searched. If parts of names are to be replaced, wildcards must be used.
- The "?" character is permitted as a wildcard. It is used to represent one character exactly. If more than one character is to be ignored, a corresponding number of "?" must be used. The "?" character is only permitted at the start of the name.
- The "*" character is permitted as a wildcard. It is used to represent any number of characters. The "*"character is only permitted in the character string that is to be searched for.
- Wildcards are only permitted in the search character string.
- There are no case distinctions.
- The section name, which is to be used as a replacement, must conform to IEC name conventions, otherwise an error message occurs.
- In accordance with IEC1131-3, only letters are permitted as the first character of item names. Should figures be required as the first character, however, the menu command **Options** → **Preferences** → **IEC extensions...** → **Allow leading digits in identifiers**.
- The specified value for the address offset is added to the corresponding address zone for located variables and direct addresses.
- The offset value is given in decimal format by default. If it is given in hexadecimal format, this can be marked as such with the prefix "16#" in front of the actual offset value (e.g. 16#100).

> **Note:** Replacing names has an effect on all variables, instance names and comments. Using wildcards runs the risk of replacing names that also happen incidentally to contain the same character string that is being searched. This would normally lead to a cancellation.

Examples of search and replace:

Replaces:	**By:**	**available names**	**Result**
Name1	Name2	Name1 Name1A NameA NameB	Name2 Name1A NameA NameB
???123	456	abc123 cde123 abcd123 abc1234	abc456 cde456 abcd123 abc1234
Name1*	Name2	Name1A XName1B NameAB	Name2A XName1B NameAB
*123	456	abc123 cde123 abcde123 abd123a	abc456 cde456 abcde456 abd123a
123	456	abc123abc cde123defghi abcde123def	abc456abc cde456defghi abcde456def
???123*	456	abc123abc cde123defghi abcde123def	abc456abc cde456defghi abcde123def

Syntax for Creating the Replace List with an External Editor

When creating the replace list using an external editor, the following syntax should also be noted:
- The replace-by string (previous name-new name) must be separated by a comma (e.g. name1,name2).
- The replace list is processed line by line. Individual replace instructions must be separated by a line break.
- The instructions for the address offset have the following structure:
  - to add an address offset:
    `<reg0>,www`
    `<reg1>,xxx`
    `<reg3>,yyy`
    `<reg4>,zzz`
  - to subtract an address offset:
    `<reg0>,-www`
    `<reg1>,-xxx`
    `<reg3>,-yyy`
    `<reg4>,-zzz`
  - Likewise, the value can be given in hexadecimal form, e.g.:
    `<reg1>,16#xxx`

Procedure for importing sections

At a Glance

In principle, sections must firstly be exported from the source project /DFB into an export file (*.sec) and then be imported by the target project/DFB. Exporting and importing from project to project or from DFB to DFB can take place in shared or in separate sessions. Exporting and importing from projects into DFBs or from DFBs into projects must take place in separate sessions.

Section export and section import

To section export a source project and then section import into a target project, the following procedure should be performed:

Step	Action
1	Open the target project in Concept.
2	Call **File** → **Export** → **Program: section(s)**.
3	In the window **Open file** select the source project, e.g. C:\SOURCE_DIR\SOURCE.PRJ
4	Select the sections to be exported from the source project.

Step	Action
5	In **Save section export under** , specify the name of the export file (*.SEC), e.g. C:\TARGET_DIR\TARGET.SEC **Reaction:** The sections are exported and saved in the *.SEC file, e.g. in TARGET.SEC. The question **Import section into project now ?** follows
6	If the question as to whether the sections should be imported is answered with **OK** , the import will be performed now. Answer **Cancel**, if you want to start the mport later, see also procedure Resuming following canceled import (See *Resuming after import cancelation, p. 647*).
7	Answer the question as to whether the project should be saved first with **OK**. **Note:** The query **Save project first ?** should be answered with **OK** , because, in the event of an import error, the current project is closed and all changes since the last save are rejected.
8	If it is required or necessary, it is possible in the table **Replace** , to make replacements for item names of variables, sections etc., as well as to define address offsets for located variables and direct addresses (refer to *Specific Changes, p. 638*).
9	Select **OK** to continue (the whole import process is canceled if **Cancel** is selected). **Reaction:** Sections, used DFBs, used derived data types and the declarations for used variables, including comments, are imported into the target project. The import is canceled and the current project closed, if ● the sections to be imported contain DFBs that are not available in the target project. ● the sections to be imported contain DFBs whose versions differ from already available DFBs. (The imported DFB version can be accepted or rejected.) ● other errors arise during import. Errors are displayed in the messages window and have to be acknowledged.
10	If the import had been canceled, eliminate the cause of the cancelation and carry out the Resuming after import cancelation (See *Resuming after import cancelation, p. 647*)procedure.

DFB export and DFB import

To section export a source DFB and to then section import into a target DFB, carry out the following procedure:

Step	Action
1	Open the target DFB in Concept DFB.
2	Call **File** → **Export** → **Program: section(s)**.
3	In the window **Open file** select the source DFB, e.g. C:\SOURCE_DIR\SOURCE.DFB
4	Select the sections to export from the source DFB.
5	In **Save section export under** specify the name of the export file (*.SEC), e.g. C:\TARGET_DIR\DFB\TARGET.SEC **Reaction:** The sections are exported and saved in the *.SEC file, e.g. in TARGET.SEC. The question **Import section into project now?** is now displayed.
6	If this question is answered with **OK**, the import is performed now. If the answer given is **Cancel**, the import is started later, refer to Resuming after import break (See *Resuming after import cancelation, p. 647*)procedure.
7	Respond to the question as to whether the project should first be saved with **OK**. **Note:** The query **Save project first ?** should be answered with **OK** , because, in the event of an import error, the current project is closed and all changes since the last save are rejected.
8	If required or necessary, it is possible in the table **Replace**, to replace item names of variables, sections etc., as well as to define address offsets for located variables and direct addresses (refer to *Specific Changes, p. 638*).
9	Select **OK** to continue (the whole import process is canceled if **Cancel** is selected). **Reaction:** Sections, used DFBs, used derived data types and the declarations for used variables, outputs and inputs are imported into the target project. The import is canceled and the current DFB closed, if ● the sections to be imported contain DFBs that are not available in the target DFB. ● the sections to be imported contain DFBs whose versions differ from already available DFBs. (The imported DFB version can be transferred or rejected.) ● other errors arise during import. Errors are displayed in the messages window and have to be acknowledged.
10	If the import had been canceled, eliminate the cause of the cancel and carry out the Resuming after import cancelation (See *Resuming after import cancelation, p. 647*)procedure.

Section export and DFB import

To section export a source project and to then section import into a target DFB, carry out the following procedure:

Step	Action
1	In Concept, delete all declarations of direct addresses and located variables in the sections to be exported. (They are not authorized in a DFB.)
2	Open the source project in Concept.
3	Call **File** → **Export** → **Program: section(s)**.
4	In the window **Open file** select the source project, e.g. C:\SOURCE_DIR\SOURCE.DFB
5	Select the sections to be exported from the source project.
6	In **Save section export under** specify the name of the export file (*.SEC), e.g. C:\TARGET_DIR\TARGET.SEC **Reaction:** The sections are exported and saved in the file *.SEC, e.g. in TARGET.SEC. The question **Import section into project now?** is now displayed.
7	Answer the question as to whether the sections should be imported with **Cancel**.
8	Close Concept.
9	Open Concept DFB and the target DFB.
10	Execute the menu command **File** → **Import** → **Program: section(s)**.
11	Select the export file (e.g. TARGET.SEC)
12	Respond to the question as to whether the project should firstly be saved with **OK**. **Note:** The question **Save project first ?** should be answered with **OK**, because in the event of an import error, the current project is closed and all changes made since the last save are rejected.
13	If required or necessary, in the table **Replace**, it is possible to replace item names of variables, sections etc., as well as to define address offsets for located variables and direct addresses (refer to *Specific Changes, p. 638*).

Step	Action
14	Select **OK** to continue (the whole import process is canceled if **Cancel** is selected). **Reaction:** Sections, DFBs used, derived data types used and the declarations of used variables, inputs and outputs are imported into the target DFB. The import is canceled and the current DFB closed, if • the sections to be imported contain DFBs that are not available in the target DFB. • the sections to be imported contain DFBs whose versions differ from those of DFBs already available. (The imported DFB version can be transferred or rejected). • other errors arise during import. Errors are displayed in the messages window and have to be acknowledged.
15	If the import had been canceled, eliminate the cause of the cancel and carry out the Resuming after import cancelation (See *Resuming after import cancelation, p. 647*)procedure.

DFB export and section import

To section export a source DFB and to then section import into a target project, carry out the following procedure:

Step	Action
1	Delete the input/output declarations in the DFB to be exported before exporting into Concept DFB, as these are not authorized in Concept projects.)
2	Open the source DFB in Concept DFB.
3	Call **File** → **Export** → **Program: section(s)**.
4	In the window **Open file** select the source DFB, e.g. C:\SOURCE_DIR\DFB\SOURCE.DFB
5	Select the sections to export from the source DFB.
6	In **Save section export under** specify the name of the export file (*.SEC), e.g. C:\TARGET_DIR\TARGET.SEC **Reaction:** The sections are exported and saved in the file *.SEC, e.g. in TARGET.SEC. The question **Import section into project now?** is now displayed.
7	Respond to the question as to whether the sections should be imported with **Cancel**.
8	Close Concept DFB.
9	Open Concept and the target project.
10	Execute the menu command **File** → **Import** → **Program: section(s)**.
11	Select the export file (e.g. TARGET.SEC).

Step	Action
12	Respond to the question as to whether the project should firstly be saved with **OK**. **Note:** The question **Save project first ?** should be answered with **OK**, because in the event of an import error, the current project is closed and all changes made since the last save are rejected.
13	If required or necessary, it is possible in the table **Replace**, to replace item names of variables, sections etc., as well as to define address offsets for located variables and direct addresses (refer to *Specific Changes, p. 638*).
14	Select **OK** to continue (the entire import process is canceled if **Cancel** is selected). **Reaction:** Sections, DFBs used, derived data types used and the declarations of variables used, incl. comments, are imported into the target project. The import is canceled and the current project closed, if • the sections to be imported contain DFBs that are not available in the target project. • the sections to be imported contain DFBs whose versions differ from those of DFBs already available. (The imported DFB version can be transferred or rejected.) • other errors arise during import. Errors are displayed in the messages window and have to be acknowledged.
15	If the import had been canceled, eliminate the cause of the cancel and carry out the Resuming after import cancelation (See *Resuming after import cancelation, p. 647*)procedure.

Resuming after import cancelation

To resume after an import cancelation, carry out the following procedure:

Step	Action
1	Open the target project/target DFB again.
2	Execute the menu command **File** → **Import** → **Program: section(s)**.
3	Select the export file (e.g. TARGET.SEC).
4	Answer the question **Back up project?:** with **Yes**. **Note:** The question **Back up project?** should be answered with **Yes**, because in the event of an import error, the current project is closed and all changes made since the last save are rejected.
5	If required or necessary, it is possible in the table **Replace**, to replace item names of variables, sections etc., as well as to define address offsets for located variables and direct addresses (refer to *Specific Changes, p. 638*).
6	Select **OK** to continue (the whole import process is canceled if **Cancel** is selected). **Reaction:** Sections, DFBs used, derived data types used and the declarations of variables used, incl. comments, are imported into the target project. The import is canceled and the current project closed, if ● the sections to be imported contain DFBs that are not available in the target project/target DFB. ● the sections to be imported contain DFBs whose versions differ from those of DFBs already available. (The imported DFB version can be transferred or rejected.) ● other errors arise during import. Errors are displayed in the messages window and have to be acknowledged.

Importing IL and ST Programs to FBD, SFC, IL or ST Sections (with Conversion)

Introduction

Using **File** → **Import** → **Program: IEC text** ASCII files with IL or ST programs can be imported to FBD, SFC, IL or ST sections. ST and IL are able to have SFC elements (when imported into SFC section). Both text languages must adhere to the grammar of IEC text languages, shown in IEC 1131-3 and in the process tables 52 56 of IEC 1131-3.

Import units

The minimum import unit is a program organization unit (POU) to IEC (PROGRAM END_PROGRAM; FUNCTION_BLOCK ... END_FUNCTION_BLOCK).

The ASCII file can contain several POUs in Concept. From one POU, one or more sections bearing the same name as the POU arise, which is provided with a current number. A new section will be begun if too little graphic space is available to store the logic. FUNCTION_BLOCK ... END_FUNCTION_BLOCK-POUs are imported as DFBs.

In Concept DFB, the ASCII file can only contain a single POU. From this POU (FUNCTION_BLOCK END_FUNCTION_BLOCK) one section arises.
Inserting POUs:

Type of POU	Import into open project	Import into open DFB
PROGRAM ... END_PROGRAM	as a section into the current project.	not possible
FUNCTION_BLOCK ...END_FUNCTION_BLOCK	as project DFB. More than one POU can be imported at the same time.	as a section into the current DFB. Only one POU can be imported.
FUNCTION ... END_FUNCTION	is changed as DFB. The function name becomes the DFB output	is changed as DFB. The function name becomes the DFB output.

Behavior in the Event of Error

In this case, sections are only stored if the ST/IL text is syntactically perfect. POUs that cannot be reproduced are ignored completely, and an error message is displayed in the message window.

Note: If the file to be imported contains more than 200 declarations (declarations of variables and FFBs, a program error is caused. If this is the case, the declarations should be divided amongst several VAR...END_VAR blocks.

Variables

The variables declared in POUs appear after the import in the Variable Editor (exceptions: SFCSTEP_STATE and SECT_CTRL type variables).

EFBs with extended parameter set

EFBs with extended parameter set (PRE_DIA, GRP_DIA, LOOKUP_TABLE, ..) are only supported up to the predefined number of inputs/outputs.

"Bracket function" with extended number of inputs

If calls from a "bracket function" with extended number of inputs, such as MUX_INT() are imported then all instances of this function work with the maximum number of inputs that occur.

Changing from IL/ST to FBD

The following restrictions occur when changing to FBD:
- The following restrictions occur when changing to FBD:
- Block items can only be called once
- only assignments and block calls
  but none:
  - RET (table 52, property 20)
  - ELSIF (table 56, property 4)
  - ELSIF (table 56, property 4)
  - CASE (table 56, property 5)
  - FOR (table 56, property 6)
  - REPEAT (table 56, property 8)
  - EXIT (table 56, property 9)
- IF not nesting (IEC 1131-3 table 56, property 4)

Changing from IL/ST to SFC

The following limitations should be noted when making a SFC import from a text file:
- Only variables are permitted as actions. Direct addresses cannot be imported.
- Only literals are allowed as time variables for identifiers.
- Transition section names are changed to standard names.
- Step monitoring times and step delay times are lost when importing.

The following additional restrictions occur when changing to SFC (table = IEC 1131-3-table):
- Transitions conditions are stored in special FBD sections (TC_secname) (table 41, property 7a ,7c, 7d). The textual import of transition conditions is not possible.
- Actions are converted into FBD sections and not linked to steps.
- no identifier SD and SL (table 45, property 8, 10), they are imported as MOVE.
- Structure components and directly addressed variables are allowed as SFC actions. This can be seen as an extension of the IEC 1131-3 standard. ST and IL exports support neither.
- Using step variables ’step.X’ , ’step.T’ cannot be imported or exported and must be generated again.

Changing from IL/ST to ST or IL

The following restrictions apply when changing to ST or IL, that were not created in Concept.

- FB, DFB and direct address declarations occur at the start of the section (VAREND_VAR)
- the source formatting (indents, comments etc) applied only to the "logic part" of the sections, i.e. no comments for declarations (VAREND_VAR), for example
- Function Block counters must be made consistent, e.g. CTU must be changed to CTU_INT
- **no** Keywords
  - TYPE_...END_TYP
  - VAR_INPUT...END_VAR
  - VAR_OUTPUT...END_VAR
  - VAR_IN_OUT...END_VAR
  - VAR_EXTERNAL...END_VAR
  - FUNCTION...END_FUNCTION
  - FUNCTION_BLOCK...END_FUNCTIONBLOCK
  - PROGRAM...END_PROGRAM
  - STEP...END_STEP
  - TRANSITION...END_TRANSITION
  - ACTION...END_ACTION
- **no** RETURN instruction (ST Editor)
- **no** RET instruction (IL Editor)

Changing to Variable Declarations

When importing variable declarations the following restrictions occur:

- No comments are imported.
- VAR_CONSTANT is imported as located variable.
  ```
  (VAR_CONSTANT
  i : INT := 10;
  END_VAR
  ```
  becomes located variable "I" with the initial value of "10")
- VAR_INPUT and VAR_OUTPUT definitions are imported into the programs as located variables (VAR).
- VAR_INPUT and VAR_OUTPUT definitions are imported into DFBs as input/output variables (VAR_INPUT, VAR_OUTPUT).

Importing (insert file) IL and ST programs into IL or ST sections

At a Glance

Using **Edit** → **Insert text file...** it is possible to import ASCII files with IL or ST programs to IL or ST sections.

This import function is a pure text import function, which can also be performed via the clipboard (cut/copy/paste). Data conversion does not take place. For this reason, the necessary variable declarations for example (also if these are contained in the ASCII file) are not automatically integrated into the Variable Editor. The necessary variable declarations must be imported explicitly via **File** → **Import** from a "variables file", or be newly created. If variable declarations are contained in the section, they must be deleted, since they generate errors in the code generation of the section. Apart from this, all information for the POU must be deleted from the program (e.g. from the export of a graphic section using **File** → **Export** → **Program: IEC text**).

Restrictions

When importing IL and ST programs the following restrictions occur:
- no keywords
  - TYPE_...END_TYP
  - VAR_INPUT...END_VAR
  - VAR_OUTPUT...END_VAR
  - VAR_IN_OUT...END_VAR
  - VAR_EXTERNAL...END_VAR
  - FUNCTION...END_FUNCTION
  - FUNCTION_BLOCK...END_FUNCTIONBLOCK
  - PROGRAM...END_PROGRAM
  - STEP...END_STEP
  - TRANSITION...END_TRANSITION
  - ACTION...END_ACTION
- VAR...END_VAR
  - only for Function Block declarations and DFBs
  - only at the start of the section for all Function Blocks and DFBs in the section
  - not for variable declarations
  - apart from this, for making direct addresses consistent: VAR %Q10:INT; END_VAR
- **no** RETURN instruction (ST Editor)
- **no** RET instruction (IL Editor)

Procedure for "Copying" an IL section from an existing project into a new project.

Procedure

To "copy" an IL section from an existing project into an IL section of a new project, perform the following steps:

Step	Action
1	Open the IL section to be exported.
2	Using **Edit** → **Save as text file...** from the menu.
3	Select a directory for the export file and give it a name. Confirm with **OK**. **Reaction:** The IL section contents are copied into a new ASCII file.
4	Execute the menu command **File** → **Export** → **Variables: Text delimited**.
5	Select the filter settings **Export variables** and **Export constants**. Select comma as the text delimiter. Confirm with **OK**.
6	Select a directory for the export file and give it a name. Confirm with **OK**. **Reaction:** The variable declarations of your project are exported to an ASCII file.
7	Using **File** → **New project** generate a new project.
8	Using **Project** → **Configuration** open the configurator.
9	Using **Configure** → **PLC type** select a PLC. Confirm with **OK**.
10	Using **File** → **New section** generate an IL section.
11	Using **Edit** → **Insert text file...** import the IL file.
12	Using **File** → **Import** → **Variables: Text delimited**(**Warning:** Text delimiter must again be comma), import the variables file into the project's Variable Editor.
13	Check the import process using **Project** → **Analyze section**. **Reaction:** The import process is now completed and the new project can be edited in the normal way (Create further sections, complete the configuration etc.)

Procedure for converting FBD sections from an existing project into IL sections of a new project.

Procedure

The process of converting FBD sections from an existing project into IL sections in a new project consists of three main steps:

Step	Action
1	Exporting FBD section (See *Exporting FBD section., p. 653*).
2	Importing FBD section into an IL section (See *Importing FBD section into an IL section, p. 654*).
3	Correcting the syntax (See *Correcting the syntax, p. 654*).

Exporting FBD section.

The procedure for exporting the FBD section is as follows:

Step	Action
1	Open the existing project.
2	Export the desired FBD section using **File** → **Export...** → **Program: IEC text**.
3	Select a directory for the export file and give it a name. Confirm with **OK**. **Reaction:** The FBD section is exported into an ASCII file.
4	Execute the menu command **File** → **Export** → **Variables: Text delimited**.
5	Select the filter settings **Export variables** and **Export constants**. Select comma as the text delimiter. Confirm with **OK**.
6	Select a directory for the export file and give it a name. Confirm with **OK**. **Reaction:** The variable declarations are exported to an ASCII file.

Importing FBD section into an IL section

The procedure for importing the FBD section into an IL section is as follows:

Step	Action
1	Using **File** → **New project** generate a new project.
2	Using **Project** → **Configuration** open the configurator.
3	Using **Configure** → **PLC type** select a PLC. Confirm with **OK**.
4	Using **File** → **New section** generate an IL section.
5	Using **Edit** → **Insert text file...** import the IL file.
6	Using **File** → **Import** → **Variables: Text delimited**(**Warning:** Text delimiter must again be comma), import the variables file into the project's Variable Editor. **Reaction:** The FBD section (in IL format) and the variable declarations were imported.

Correcting the syntax

The procedure for correcting the syntax is as follows:

Step	Action
1	Delete the line `PROGRAM`. (It contains the name of the old project.)
2	Delete any lines between `VAR` and `END_VAR` which do not contain Function Block or DFB declarations (e.g. variable declarations).
3	Delete all lines from `INITIAL_STEP` to `END_STEP`. (They contain the sections processing sequence of the old project.)
4	Change the `ACTION` lines to comment lines, e.g. `(* ACTION xxx *)`. (They contain the names of the FBD sections.)
5	Delete the `END_ACTION` line.
6	Delete the `END_PROGRAM` line.
7	Verify the import process using **Project** → **Analyze section** and correct any errors. **Reaction:** The import process is now completed and the new project can be edited in the normal way (Create further sections, complete the configuration etc.)

21.5 Variables import

At a Glance

Overview

This section describes the importing of variables.

What's in this Section?

This section contains the following topics:

Importing Variables in "Text Delimited" Format

Introduction

Using **File** → **Import** → **Variables: Text Delimited**, the variable declarations can be imported from an ASCII file into the variable editor in text delimited format.

Importing Initial Values

Initial values of variables in derived data types cannot be imported with this import format. If you wish to import initial values of variables in derived data types, select the IEC text import export/import format.

General Format Description

An ASCII file in "text delimited" format must conform to the following conditions:
- The character set used conforms to ANSI (Windows).
- The parameters of a variable are executed within one line.
- The individual parameters are separated from one another by a user-defined character.
- Leading and following spaces are allowed in any field (Exception: if a space has been used as a separator), the import function deletes the latter (with the exception of the comment field).
- The selected separator must not be contained in the individual parameters.
- Concept is not case-sensitive, in accordance with IEC name conventions. This should be adhered to for variable names.
- Overlapping between pre-existing addresses and addresses to be imported can be prevented in the following way: in the **Options** → **Preferences** → **Analysis...** → **Analysis Preferences** dialog, activate the **Treat Overlap of Addresses as an Error** option.

Order of Parameters within a Line

Order of Parameters within a Line:
- Variable flag
- Variable name (symbolic name)
- Data type
- Hardware address
- Initial value
- Comment

Meaning of Variable Flags

Possible values for the variable flags are:
- 0 or N= the symbolic name refers to a non-exportable variable
- 1 or E= the symbolic name refers to an exportable variable
- 2 or C= the symbolic name refers to a constant
- 3 or I = the symbolic name refers to an Input (See *Formal parameters, p. 427*) (Concept DFB only)
- 4 or O = the symbolic name refers to an Output (See *Formal parameters, p. 427*) (Concept DFB only)
- 5 or M = the symbolic name refers to a VARINOUT variable (See *Combined Input/Output Variables (VARINOUT Variables), p. 428*) (Concept DFB only)
- S = Structured variable, see *Importing structured variables, p. 659*.

Only variables with the 0/N or 1/E variable flag value are imported as located variables. All others are imported as unlocated variables.

If the variable flag is set at 2/C, the hardware address is ignored.

The values 3/I and 4/O are only permitted in Concept DFB. In this case, the values of the address fields are used for the position of the corresponding inputs and outputs. The variable flag value 1/E is imported into Concept DFB as variable flag value 0/N.

Structure of the Hardware Address Field

Structure of the Hardware Address Field (Example: %4:100):
- Characters for direct addresses "%" (may be missing)
- Address type
  - 0 = output, discrete
  - 1 = input
  - 3 = input word
  - 4 = output word, discrete word
- Separator ":" or ".".
  If no separator is used, the address must be 6 characters long.
- Address

Examples of an Address Description

Output register 123 :
- %400123 or
- %4.123 or
- %4:123 or
- 400123 or
- 4.123 or
- 4:123

IEC Address Conventions

The IEC address conventions can also be used (e.g. %QX100 corresponds to 000100):

Address Type	Concept Designation	IEC Designation
Output, discrete	0x	%QX,%Q
Input	1x	%IX,%I
Input register	2x	%IW
Output register, discrete register	3x	%QW

Empty Fields

Empty fields are represented by two consecutive separators.
The following fields are allowed to be empty:
- Hardware address
- Initial value
- Comment

Missing Fields

The following fields are allowed to be missing:
- Comment
- Comment and initial value
- Comment and initial value and hardware address

Importing structured variables

At a Glance

The basic structure of the file corresponds to that of the variables in text delimited (See *Importing Variables in "Text Delimited" Format, p. 656*) format.

Additional usage designations

In addition, the following points should be taken into account:
- Multiple rows are necessary to describe a variable.
- Each of these rows must correspond to the format of variables in delimited text format.
- A structured variable with initial values is described by an introducing row with the following structure:
  1. Variable flag
  2. Variable name (symbolic name)
  3. Name of derived data type
  4. Hardware address
  5. Empty field
  6. Comment
- This introductory line is followed by at least one component description. This component description results from the description of the element components (element data type) in the form of a row with the following structure (a component does not have to be described if its initial value is the same as the standard value). The sequence in which the individual components are executed is insignificant.
  1. Character "S" (S stands for structured)
  2. Path of components (the variable name does not have to be included)
  3. Field for IEC data type (this field can remain empty)
  4. Empty field
  5. Initial value
  6. Empty field

Component description error trapping

Component description error trapping
- If a variable component is described more than once, the last description is used.
- If the specified component is not contained in the currently described variable, the component description is ignored and a warning is given.
- If the field for the components path is empty, the component description is ignored and a warning is given.
- If the field for the IEC data type is not empty, the specified data type is checked. If the specified data type and the data type of the component are not the same, the component description is ignored and a warning is given.
- Entries in the address field are ignored.
- Entries in the address field are ignored.

Example: Structured variables in "Text delimited" format

Structured data type definition ESI_IN:

```
ESI_In:   (* ESI - input data *)
  STRUCT
    in:          ESI_InOut;     (* ESI input data *)
    esi:         ESI_Status;
    dummy:       BYTE;          (* supplement to modulo 16 *)
    slot:        Exp_Status;
  END_STRUCT;

ESI_InOut:  (* ESI input / output data structure *)
  STRUCT
    tstat:       BYTE;     (* transfer status, handshake *)
    blocks:      BYTE;     (* number of used blocks *)
    res:         BYTE;     (* reserved *)
    block:       ESI_BlockArr14;     (* data block *)
  END_STRUCT;

ESI_BlockArr14: ARRAY[1..14] OF ESI_Block;

ESI_Block:  (* datas of ESI *)
   STRUCT
    func:        BYTE;     (* function *)
    mux:         WORD;     (* distribution *)
    attr:        BYTE;     (* attribute *)
    cause:       BYTE;     (* reason *)
    station:     WORD;     (* station number *)
    object:      WORD;     (* objekt number *)
    data:        ByteArr9; (* data bytes *)
   END_STRUCT;

ByteArr9:        ARRAY [1..9] OF BYTE;    (* 9 bytes *)

ESI_Status:  (* Status of ESI *)
   STRUCT
    wdog:        BYTE;     (* expert watchdog-counter *)
    stat1:       BYTE;     (* error status 1 *)
    stat2:       BYTE;     (* error status 2 *)
    stat3:       BYTE;     (* error status 3 *)
    slot:        WORD;     (* slot number *)
    user:        WORD;     (* virtual slot number *)
    esitime:     DPM_Time; (* time stamp *)
  END_STRUCT;
```

```
DPM_Time:  (* time stamp *)
  STRUCT
    sync:         BOOL;      (* sync clock *)
    ms:           WORD;      (* milli-seconds *)
    min:          BYTE;      (* minutes *)
    hour:         BYTE;      (* hours; (hour AND 16#80) *)
                             (* = day light saving time *)
    day:          BYTE;      (* days of week *)
    mon:          BYTE;      (* month *)
    year:         BYTE;      (* year *)
  END_STRUCT;

STRUCT
  Exp_Status:  (* error status of transfer *)
    ErrFlag1:     BOOL;      (* TRUE: epxert not pluged *)
    ErrFlag2:     BOOL;      (* TRUE: Bit 7 of DPM *)
                             (* Identcode is set; *)
                             (* logical DMP-access-error *)
    UserStatus:  WORD;      (* status of expert *)
    ErrNo:        WORD;      (* errornumber *)
END_STRUCT;
```

Representation of variables "demo" of ESP_IN data type in delimited text format:

```
1;demo;ESI_In;400002;;structured data type
S;in.tstat;BYTE;;16#0F;
S;in.blocks;BYTE;;16#0F;
S;in.res;BYTE;;16#0F;
S;in.block[1].func;BYTE;;16#0F;
S;in.block[1].mux;WORD;;16#000F;
S;in.block[1].attr;BYTE;;16#0F;
S;in.block[1].cause;BYTE;;16#0F;
S;in.block[1].station;WORD;;16#000F;
S;in.block[1].object;WORD;;16#000F;
S;in.block[1].data[1];BYTE;;16#0F;
S;in.block[1].data[5];BYTE;;16#0F;
S;in.block[3].func;BYTE;;16#0F;
S;in.block[3].mux;WORD;;16#000F;
S;in.block[3].func;BYTE;;16#0F;
S;in.block[3].cause;BYTE;;16#0F
S;in.block[3].station;WORD;;16#000F
S;in.block[3].object;WORD;;16#000F
S;in.block[3].data[1];BYTE;;16#0F
S;in.block[3].data[2];BYTE;;16#0F
S;esi.wdog;BYTE;;16#0F
```

```
S;esi.stat1;BYTE;;16#0F
S;esi.stat2;BYTE;;16#0F
S;esi.stat3;BYTE;;16#0F
S;esi.slot;WORD;;16#000F
S;esi.user;WORD;;16#000F
S;esi.esitime.sync;BOOL;;TRUE
S;esi.esitime.ms;WORD;;16#000F
S;esi.esitime.min;BYTE;;16#0F
S;esi.esitime.hour;BYTE;;16#0F
S;esi.esitime.day;BYTE;;16#0F
S;esi.esitime.mon;BYTE;;16#0F;
S;esi.esitime.year;BYTE;;16#0F;
S;dummy;BYTE;;16#0F;
S;slot.ErrFlag1;BOOL;;FALSE;
S;slot.ErrFlag2;BOOL;;FALSE;
S;slot.UserStatus;WORD;;16#000F;
S;slot.ErrNo;WORD;;16#000F;
```

Importing variables in Factory Link format

Description

Using **File** → **Import** → **Variables: Factory Link** variable declarations in Factory Link format can be imported. In addition, carry out a Factory Link export and specify the Factory Link version when importing into Concept.

If your Factory Link version of Concept is not supported, please call our hotline.

Note: Factory Link is case-sensitive with variable names. Concept does not differentiate in accordance with IEC naming conventions. This should be adhered to during import

21.6 Import/Export of PLC Configuration

Introduction

Overview

This Section describes the import and export of the PLC configuration with Concept or Concept Converter.

What's in this Section?

This section contains the following topics:

Import/Export of PLC Configuration using Concept

Introduction

Using the Import/Export function a PLC configuration can be exported out of a current (open) project and subsequently re-imported.

Config. Export and Config. Import

To export and subsequently import SPS configurations, proceed as follows:

Step	Action
1	To export the PLC configuration from the current project, start Concept, open the desired project and select **File** → **Export** → **Configuration**.
2	In the **Folder** field, select the target directory for the PLC configuration to be exported.
3	In the **File name** field, enter a name for the Export file (NAME.CCF) and press **OK**. **Response:** The PLC configuration is stored in the selected directory as an ASCII file.
4	To import a PLC configuration into a project, open the desired project.
5	In Concept select the **File** → **Import** → **Configuration**menu command.
6	From the **File type** list select the **Concept Configuration entry. (*.CCF)**.
7	In the **Folder** field, select the desired directory.
8	From the **File name** list select the PLC configuration to be imported (NAME.CCF) and click on **OK**.
9	**Warning:** The current PLC configuration of the chosen project will be overwritten. Answer the question with **OK**. **Response:** The PLC configuration is imported.

Import/Export of PLC Configuration using Concept Converter

Introduction

The Concept Converter's import/export function enables you to export the configuration from one project (Project A) and import it into another project (Project B).

Config Export and Config Import

In order to export and then import a PLC configuration, carry out the following steps:

Step	Action
1	To export the PLC Configuration from project A, start the Concept Converter and select **File** → **Export** → **Configuration**.
2	From the **Folder** field, select the Project A system directory.
3	Select the PLC configuration to be exported (PROJECTNAME.C1) and click on **OK**. **Response:** The PLC configuration is filed in the system directory in the form of an ASCII file (PROJECTNAME.CON).
4	To import the PLC configuration into Project B, copy the exported file into the system directory of Project B.
5	In Concept Converter select the **File** → **Import** menu command.
6	From the **File Type** list box select the **Configuration (*.CON)** entry.
7	From the **Folder** field, select the Project B system directory.
8	From the **File Name** list box, select the PLC configuration (PROJEKTNAME.CON) to be imported and click on **OK**.
9	**Caution:** The current PLC configuration of the selected project will be overwritten. Answer the question with **OK**. **Response:** The PLC configuration will be imported.

Documentation and Archiving

22

At a Glance

Overview

This chapter describes the documentation, the archiving and deleting of projects, DFBs and macros.

What's in this Chapter?

This chapter contains the following sections:

22.1 Documentation of projects, DFBs and macros

At a Glance

Overview

This section describes the documentation of projects, DFBs and macros.

What's in this Section?

This section contains the following topics:

Documentation contents

At a Glance

The contents of the documentation can range in length from one on-screen page to the entire documentation of a project. The order in the first chapter is given as in the dialog box **File** → **Print** → **Documentation contents** and cannot be changed.

Project documentation

The following chapter can be printed for project documentation using the menu command **File** → **Print**:
- Project description
- Derived data types
- Using state RAM
- State RAM values
- Using the DFBs
- Using the EFBs
- PLC configuration
- I/O Map
- Execution sequence of the sections
- Project structure
- Messages
- ASCII messages only with Concept for Quantum
- Variable lists
- Use of variables
- Contents of sections
- Contents directory for the printed documentation

DFB/macro documentation

The following chapter can be printed for DFB/macro documentation using the menu command **File** → **Print** :
- DFB/macro description
- Derived data types
- Using the DFBs
- Using the EFBs
- Execution sequence of the sections
- Messages
- Variable lists
- Use of variables
- Contents of the sections
- Contents directory for the printed documentation

Documentation Layout

Print Format

The printout can be in either portrait or landscape mode. This is set up in the dialog box **File** → **Printer Setup** → **Select Printer**.

Page Numbering

The pages are numbered linearly. The starting page number can be selected by the user.

Page Size

The left margin is 12 characters wide. The area for text and graphics is approximately 132 characters wide, the height depends on the header and footer files. If the header and footer files are not activated or the keyword "%PAGENO" is not contained in them, the page number will be printed automatically in the bottom right corner of the page.

Page Breaks

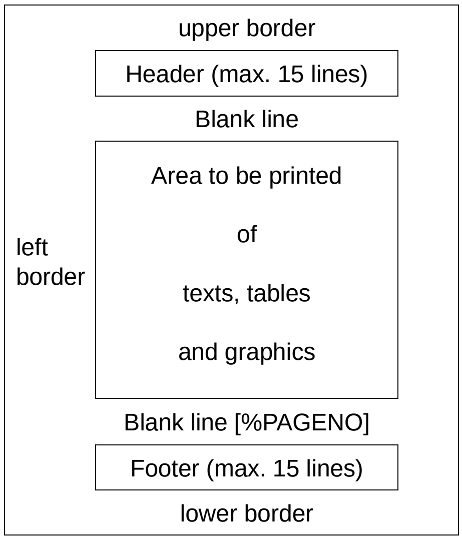
If a graphics section does not fit on a printed page, the section will be divided - like a map - in the printout. In this case page references are printed in all four corners of the graphics area to show which page the graphics are continued on. The **View** → **Page Break** menu option displays the page break corresponding to the printer set in **File** → **Printer Setup** and to the enlargement factor in the editor window.
Also see the *Defining Page Breaks for Sections, p. 673* description.

Size and Fonts

In text sections the font size in the printout cannot be altered. Emphasis of keywords is represented in the printout using bold and italic typefaces.

Standard Layout

Standard Layout:



Header

It is possible to give your documentation a header. The header is stored as an ASCII file and can be created using any ASCII editor. The maximum file size is 15 lines or approx. 2 Kbytes.

A sample file called "HEADER.TXT" is available in the Concept directory. This file can be modified as required. Keywords (See *Use of keywords, p. 676*) can also be used with it.

Footer

It is possible to give your documentation a footer. The footer is stored as an ASCII file and can be created using any ASCII editor. The maximum file size is 15 lines or approx. 2 Kbytes.

An sample file called "FOOTER.TXT" is available in the Concept directory. This file can be modified as required. Keywords (See *Use of keywords, p. 676*) can also be used with it.

Front Page

It is possible to give your documentation a front page. The front page is stored as an ASCII file and can be created using any ASCII editor. The size of the file is unlimited. An sample file called "FRONTPG.TXT" is available in the Concept directory. This file can be modified as required. Keywords (See *Use of keywords, p. 676*) can also be used with it.
The printout of the front page also contains the header and footer if these are switched on.

Defining Page Breaks for Sections

Introduction

For printing graphics in FBD, LD and SFC sections, you can define the values for the page break and paper orientation of the graphics. The higher the value you select, the smaller the graphics will be displayed. But in return more space is available on a page.

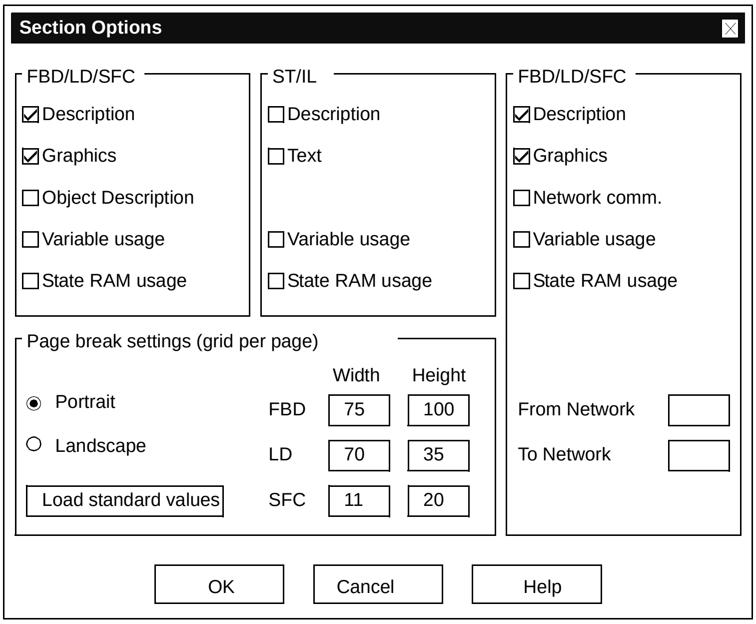
Settings

You can set the values for the page break for portrait and landscape. When changing the paper format, the settings for the other format stay saved. Using the **Download standard values** command button, the standard values from the CONCEPT.INI file can be loaded.
When defining values for the width and height of the paper, you should make sure that the different editors show different grid units.
The min. and max. values are:

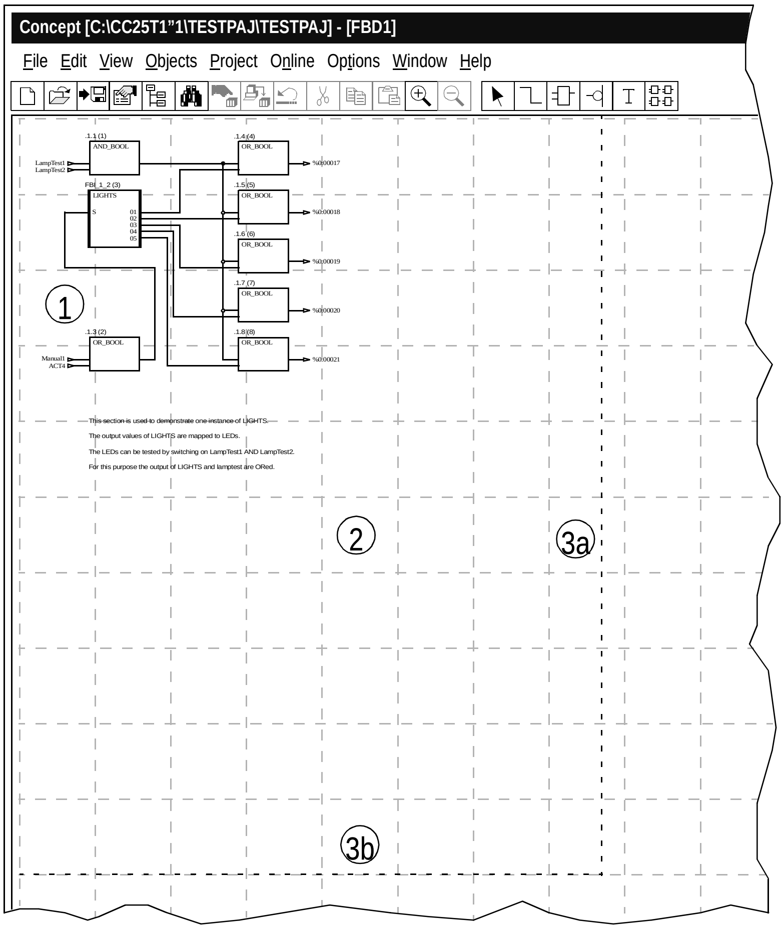
Section	1 grid unit equals the value	Paper Width	Paper Height
FBD	10	30 - 300	30 - 230
LD	8	30 - 400	10 - 230
SFC	1	4 - 32	4 - 60

Example for FBD section

Dialog setting



Representation in the FBD editor window



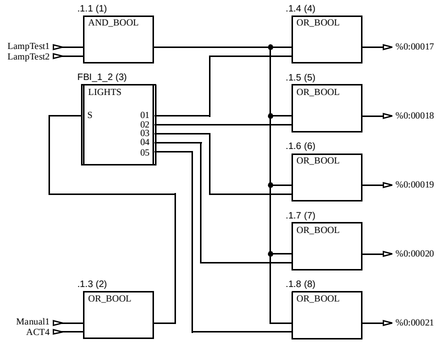
1 FBD section

2 Grid view (View -> Grid)

3a Page break, width: 75 (View -> Page break)

3b Page break, height: 100 (View -> Page break)

Print-out



This section is used to demonstrate one instance of LIGHTS.

The output values of LIGHTS are mapped to LEDs.

The LEDs can be tested by switching on LampTest1 AND LampTest2.

For this purpose the output of LIGHTS and lamptest are ORed.

Use of keywords

At a Glance

Keywords allow project or object specific information to be inserted into the header, footer and title page files.

Usable keywords

Table of usable keywords:

%PROJNAME	**Project name**
%SECTNAME	Section name
%VERSION	Program/DFB version
%CREDATE	Creation date
%MODDATE	Date of last project/DFB modification
%DATE_D	Current date (European format, DD.MM.YY)
%DATE_US	Current date (US format, DD.MM.YY)
%PAGENO	Current page numbers
%RECT(Column,Width,Height)	Draws a rectangle with its top left-hand corner in the current line
%HLINE(Column,Length)	Draws a horizontal line in the current line
%VLINE(Column,Length)	Draws a vertical line starting in the current line

Note: The total number of lines in the header, footer or title page file must agree with the number of lines needed to print rectangles and vertical lines.

Example: Header with keywords

Contents of the ASCII file:

```
%RECT (1,132,4)    %VLINE (24,4)    %VLINE (110,4)
     S A              Project comment            Name
   CONCEPT                                         %DATE_D
¶
```

> **Note:** The symbol ¶ is not entered, it should only show that the file ends with a blank line.

Expression:

S A CONCEPT	Project comment	Name 01.04.99

22.2 Managing projects, DFBs and macros

At a Glance

Overview

This section describes the archiving and deletion of projects, DFBs and macros.

What's in this Section?

This section contains the following topics:

Archiving projects, used DFBs, EFBs and data type files

At a Glance

When archiving projects, used DFBs, EFBs and data type files, all data of the project is collected and compressed. The *.PRZ file is created and put into the same directory as the project itself. The file can be decompressed at any time thereafter.

Archiving Projects

The procedure for archiving projects is as follows:

Step	Action
1	Start Concept. **Note:** No project may be open during the archiving procedure, otherwise the **Archiving...** command cannot be selected.
2	To archive, select **File** → **Archiving...**. **Reaction:** A window showing the Concept projects appears.
3	Select the project to be archived from the window and press **OK**. **Reaction 1:** A check for whether a compressed *.PRZ file has the same name is performed. If there is a file with the same name you are requested to confirm whether the existing file should be replaced by the new file. **Reaction 2:** The project data is compressed and saved in the *.PRZ file and is then found in the same directory as the project.

Unpacking Archived Projects

The procedure for unpacking archiving projects is as follows:

Step	Action
1	Select **File** → **Open**. **Reaction:** A window showing all Concept projects appears.
2	Go to the list field **File Type** and select option **Archived Projects (*.prz)**. **Reaction:** The archived Concept projects are displayed.
3	Select the project that you want to open and press **OK**. **Reaction 1:** A check for whether a *.PRJ file has the same name is performed. If there is a file with the same name you are requested to confirm whether the existing file should be replaced by the new file. **Reaction 2:** A check for whether DFBs, EFB libraries and data type files with the same name exist, is performed. If there is a file with the same name you are requested to confirm whether the existing file should be replaced by the new file. **Reaction 3:** The **Archive content** dialog is opened.
4	Select **Unpack**. **Reaction 1:** The project data is decompressed and stored as a normal Concept project. The project is then found in the same directory as the archived file. **Reaction 2:** The project is automatically opened in Concept.
5	Establish a connection between the PC and the PLC with **Online** → **Connect...**. **Reaction:** The PC and PLC are found in the same status as before the archive procedure.

Archiving/ unpacking global DFBs

When archiving or unpacking the used global DFBs, the following sequence should be used:

Step	Action
1	The project directory is searched for an existing GLB directory.
2	The relevant settings are checked in the CONCEPT.INI file. For example: [Path]: GlobalDFBPath=x:\DFB [Upload]: PreserveGlobalDFBs=0 In this example, the DFB directory of the path defined is searched for global DFBs.
3	The DFB directory in x:\CONCEPT\DFB is searched.

Only the global DFBs from one directory are used or only put into one directory, i.e. if step 1 is unsuccessful, then step 2 follows, step 3 is only performed if neither of the first two are successful.

Diagnostic Information

When downloading the project, diagnostic information is created and put into the corresponding directory. Then, the status between PC and PLC becomes EQUAL. When archiving the project, this diagnostic information is compressed with the other project data and stored in a file.
To use the diagnostic information after it is decompressed, make sure that the status between the PC and the PLC is EQUAL when archiving. Downloading is no longer required and diagnostics can be run immediately.
If the status is anything else between the PC and the PLC, e.g. NOT EQUAL, then this status will be displayed while decompressing and after the connection (**Online** → **Connect...**). Downloading is therefore required to put the system into operation. Downloading also creates new diagnostic information while the old information is lost however.

Deleting projects, DFBs and macros

Deleting projects, DFBs and macros

The procedure for deleting projects, DFBs and macros is as follows:

Step	Action
1	Delete the project/DFB/macro directory (including the subdirectory "dfb"). If only certain DFBs/macros need to be deleted from this directory, open the subdirectory and delete all files with the required DFB/macro name (name *).
2	Use global DFBs, and global macros in the project/DFB and if these also need to be deleted, they must be deleted separately. Open the subdirectory "dfb" of the Concept directory and delete all files carrying the name DFBs/macros (name *).

Simulating a PLC

23

Preview

Overview

This chapter describes how to simulate a PLC. By using a simulator the functions of a program may be tested without the actual required hardware.

What's in this Chapter?

This chapter contains the following sections:

23.1 Simulating a PLC (16-bit simulator)

Simulating a Controller

Introduction

This section describes the 16-bit simulator Concept SIM.

Area of Application

Concept SIM may be used to simulate any PLC (Quantum, Compact, Momentum, Atrium) in order to test the user program "online" without hardware.

The simulator is only available for the IEC languages (FBD, SFC, LD, IL and ST).

The 16-bit simulator Concept SIM is used for testing programs containing Concept EFB generated 16-bit EFB.

Note: If your program does not contain 16-bit EFBs created with Concept EFB, you should use the 32-bit simulator (PLCSIM) to simulate a PLC.

Max. Number of Variables

When using the 16 bit simulator Concept SIM, a specific number of state RAM references (**Project** → **PLC configuration** → **Configuring** → **Memory Partitions**) may not be exceeded.

The table below shows the maximum number of these state RAM references:

Reference type	max. number
0x	60000
1x	5008
3x	4000
4x	24000

Concept vs. Concept SIM

Concept SIM and Concept may only be opened independently, i.e. when starting Concept SIM, Concept cannot be open. It is therefore advisable to decide before starting Concept, whether the simulator or the controller should perform the test. In each case, make sure that the simulator is turned on or off as required.

Activating Concept SIM

The procedure for activating Concept SIM is as follows:

Step	Action
1	Close Concept if it is open.
2	Open Concept-SIM by double-clicking on the Concept-SIM icon.
3	Click on the **File** main menu and activate the **Simulation on** menu command. **Response:** The simulator is on.
4	Exit Concept SIM via the **File** main menu using the **Exit** menu command.
5	Start Concept.
6	From **Online** → **Connect...** open the **Connect to PLC** dialog window.
7	For **Protocol type:** always select **Modbus Plus**, even if your real PLC will be coupled via a different bus at a later stage. **Response:** The simulator will now be displayed as a PLC in the node list of the Modbus Plus network.
8	Now create a link to the simulated PLC by double clicking on the list entry or via **OK**. **Response:** You may now test the behavior of your IEC user program.

Note

Note: Please note that the simulator remains active even after rebooting the PC. To build a link to a PLC the simulator must be explicitly terminated.

Disabling Concept SIM

The procedure for disabling Concept SIM is as follows:

Step	Action
1	Close Concept if it is open.
2	Open Concept-SIM by double-clicking on the Concept-SIM icon.
3	Click on the **File** main menu and select the **Simulation Off** menu command. **Response:** The simulator is on.
4	Exit Concept SIM via the **File** main menu using the **Exit** menu command.

23.2 Simulating a PLC (32-bit simulator)

At a Glance

Overview

This Section describes how to simulate a PLC with the 32-bit simulator Concept PLCSIM32.

What's in this Section?

This section contains the following topics:

Concept-PLCSIM32

Introduction

The Concept-PLCSIM32 program simulates any PLC unit (Quantum, Compact, Momentum, Atrium) and its signal states.

Area of Use

The simulator is presently only available for IEC languages (FBD, SFC, LD, IL and ST).

Note: Not supported:
- LL984 language
- Loadables, e.g. ULEX
- 6x-Register (extended memory)
- RIO
- DIO
- Backplane Expander

Note for Windows 98 and Windows NT

Since the simulator is connected to Concept via a TCP/IP link, you need a card in your computer to handle the TCP/IP interface (when using Windows 98 or Windows NT). If your computer is not equipped with such a card, it can be simulated. Follow the procedure described in Simulation of a TCP/IP interface card in Windows 98 (See *Simulating a TCP/IP interface card in Windows 98, p. 692*) or Simulation of a TCP/IP interface card in Windows NT (See *Simulating a TCP/IP interface card in Windows NT, p. 693*).

When using Windows 2000, simulating a TCP/IP interface card is not necessary because all drivers needed for Concept PLCSIM32 are installed automatically.

Structure of the dialog box

The title bar shows the name of the application (PLC Sim32) and the address of your PC-interface card.

The first text box in the simulator window shows the status of the simulated PLC. This field is read-only. The displayed status is determined by Concept, as with a real PLC.
The status may be shown as the following:
- **DIM** (Dim Awareness)
  The simulator is in an undefined state.
- **STOPPED**
  The simulator (the simulated PLC) is stopped.
- **RUNNING**
  The simulator (the simulated PLC) is running.

The type of PLC to be simulated can be selected from the first list box.

The following registers are available:
- **State RAM**
  Provides an overview of the signal memory.
- **I/O Modules**
  Shows the configuration currently loaded or the signal memory of a selected group of components.
- **Connections**
  Displays connections between the simulator and programming device(s).

Simulating a PLC

Overview

A controller is simulated with the PLCSIM32 simulator using 4 main steps:

Step	Action
1	Program creation and controller configuration.
2	Activating the simulator.
3	Construction of the connection between Concept and simulator.
4	Downloading the program.

Program creation and controller configuration

The following steps describe how to create programs and configure the controller.

Step	Action
1	Create your program and your controller configuration in Concept.
2	Save your project with **File** → **Save**.

Activating the simulator

The following steps describe how to activate the simulator:

Step	Action
1	Run PLCSIM32 simulator in the Concept program group.
2	Select the controller type appropriate to your project in the simulator.

Construction of the connection

The following steps describe how to construct the connection between Concept and the simulator.

Step	Action
1	Using **Online** → **Connect...** open the **Connect to PLC** dialog in Concept.
2	Select the **IEC Simulator (32-Bit)** entry in the Protocol Type list box.
3	In the **Access** range, activate the **Change configuration** option button.
4	Confirm with **OK**. **Response:** A connection has been made between the programming unit and the simulator. A note then appears, saying that the configurations of the programming unit and the simulator are different.

Downloading the program

The following steps describe how to download the program:

Step	Action
1	Using **Online** → **Download** open the **Download Controller** dialog.
2	Confirm with **Download**. **Response:** Your program and your configuration are loaded into the simulator. You will be asked if you wish to start the controller.
3	Confirm with **Yes**. **Response:** You may now test the behavior of your IEC user program.

Simulating a TCP/IP interface card in Windows 98

Introduction

As the coupling between Concept and the simulator PLCSIM32 is made via a TCP/IP coupling, a TCP/IP interface card is needed in the PC. If your PC does not have one of these cards, it may be simulated.

> **CAUTION**
>
> **Risk of PC problems**
>
> Do **NOT** complete this procedure if your PC already has a TCP/IP connection. The software installation of the TCP/IP connection would be destroyed by this procedure. Only carry out this procedure once, otherwise PC problems may arise.
>
> **Failure to follow this precaution can result in injury or equipment damage.**

Simulating a TCP/IP Interface Card

Carry out the following steps to simulate a TCP/IP interface card in Windows 98:

Step	Action
1	In Windows 98 invoke **Start** → **Settings** → **Control Panel**.
2	From **Software** open software settings.
3	From the **Windows Setup** register select the **Links** entry and click on the **Details...** command button.
4	Check the **DFU network** entry here and confirm with **OK**. (To do this, you may require the Windows system CD.) **Response:** The computer reboots. The DFU network and the TCP/IP protocol are available to the system after the reboot. (Concept can only connect to the simulator.)

Simulating a TCP/IP interface card in Windows NT

Introduction

As the coupling between Concept and the simulator PLCSIM32 is made via a TCP/IP coupling, a TCP/IP interface card is needed in the PC. If your PC does not have one of these cards, it may be simulated.

	CAUTION
⚠	**Risk of PC problems** Do **NOT** complete this procedure if your PC already has a TCP/IP connection. The software installation of the TCP/IP connection would be destroyed by this procedure. Only carry out this procedure once, otherwise PC problems may arise. **Failure to follow this precaution can result in injury or equipment damage.**

Simulating a TCP/IP Interface Card

The main steps for simulating a TCP/IP interface card in Windows NT are as follows:

Step	Action
1	Setting the basic settings.
2	Installing a new modem.
3	Setting the workgroup.

Setting the Basic Settings

The procedure for setting the basic settings is as follows:

Step	Action
1	In Windows NT, invoke **Start** → **Settings** → **Control Panel** → **Network** and answer **Yes** to the question. **Response:** The **Network Setup Wizard** dialog is opened.
2	Deactivate the **Wired to the network** option.
3	Select the **Remote access to the network** option. **Response:** The network card installation dialog will be opened.
4	Click on **Next** (without installing a network card). **Response:** The dialog for selecting a network protocol will be opened.
5	Select the **TCP/IP-Protocol** option.
6	Deactivate all the other options and click on **Next**. **Response:** The dialog for selecting services will be opened.
7	Click on **Next** (without making any changes in the dialog).
8	Answer the question with **Next**. Response: The **Windows NT Setup** dialog is opened.

Installing a New Modem

The procedure for installing a new modem is as follows:

Step	Action
1	Insert the Windows NT CD and specify the path for the installation data files (for example `D:\i386`). Click on **Resume**. **Response:** The **TCP/IP Setup** dialog is opened.
2	Click on **No**. **Response:** The **Remote Access Setup** dialog is opened.
3	Click on **Yes**. **Response:** The **Install New Modem** dialog is opened.
4	Select the **Don't detect my modem; I will select it from a list.** option and press **Next**. **Response:** The dialog for selecting a modem is opened.
5	Select a standard modem (for example **Standard 28800 bps modem**) and press **Next**. **Response:** The dialog for selecting the connection is opened.
6	Select the **Selected ports** option and the **COM** interface. Click on **Next**. **Response:** The **Standard information** dialog is opened.
7	Select your country.
8	Enter the code for your town (your area code) and click on **Next**. **Response:** The **Install New Modem** dialog is opened.
9	Click on **Finish**. **Response:** The **Add Remote Access Setup device** dialog is opened.
10	Click on **OK**. **Response:** The **Remote Access Setup** dialog is opened.
11	Click on **Next**. **Response:** The **Network installation assistant** dialog is opened.
12	Click on **Next** twice. **Response:** The dialog for setting the workgroup is opened.

Setting the Workgroup

The procedure for setting the workgroup is as follows:

Step	Action
1	Select the **Workgroup** option and enter the `WORKGROUP` name. Click on **Next**.
2	Click on **Finish**. **Response:** The **Network Settings Changes** dialog is opened.
3	Click on **Yes** to restart. **Response:** Your PC now simulates a TCP/IP network and the 32-bit simulator PLCSIM may be used.

Concept Security

24

At a Glance

Overview

This chapter describes Concept Security.

What's in this Chapter?

This chapter contains the following topics:

General Description of Concept Security

At a Glance

You can define access rights (See *Access Rights, p. 700*) (user definitions) using Concept Security. The access rights limit the functionality of Concept and its utilities for certain users.

> **Note:** The Editor LL984 cannot be protected with Concept Security.

Projects/DFBs can be protected (See *Protecting Projects/DFBs, p. 709*) from being edited using Concept Security.

Scope

The access rights defined for a user are valid for all projects within the Concept installation. If a user edits projects in different Concept installations, he has to be defined as a user in each Concept installation.

Max. number of users

A maximum of 128 users can be defined.

Activating Concept Security

After Concept is installed, Concept Security is inactive and must be activated by the system administrator (Supervisor).

The system administrator

Access rights are defined and Concept Security is switched on/off by the system administrator (user name: **Supervisor**).
When Concept is installed, a password file is automatically created for the "Supervisor" (system administrator) with an empty password. This user has "Supervisor" access rights.

Changing the access rights online

Concept Security and Concept/Concept-DFB can be started at the same time, i.e. the access rights can be changed while Concept/Concept-DFB is running and become active immediately.

Creating a log

In Concept, if you go into the **Options** → **Preferences** → **Common...** → **Common Preferences** dialog box in the **Logging** area and activate the **File** option (and enter a path name), the log function is activated. A file with the name YEARMONTHDAY.LOG (e.g. 19980926.LOG) is created in the folder you selected, which contains a log of all system critical (runtime relevant) changes.
The following data (and other data) is logged in the ASCII file:
- Section name
- EFB/DFB instance name, FB type name
- Pin name
- [variable name] [literal] [address]
- Old value
- New value
- User name (if password protection is activated in Concept Security)
- Date and time (also see **Options** → **Preferences** → **Common...**)

The following logging can be carried out during log-on:
- Modification of the user rights
- Deleted user
- Aborted log-on

In Concept, you can view the current log using the menu command **File** → **View Logfile**.

Encrypt Logfile

Logging write access on the PLC can be stored in an encrypted YEARMONTHDAY.ENC file (e.g. 20021025.ENC). To do this, go to the **Project Properties** (main menu **Project**) dialog box and activate the control box **Secure Application**. In Concept, you can view the current log using the menu command **File** → **View Logfile**. If the current log is encrypted, the content of the ENC file is automatically opened in a view tool and can viewed or printed there. Supervisor rights are required to do this.

Access Rights

At a glance

The access rights are set up in a hierarchy; if the user has the rights for a certain level, he also has the rights to all lower levels.

Access Right Levels

The following levels are defined (from lowest to highest):

Level	Access rights	Assigned Functionality
1	Read only	The user can view projects offline and online, but cannot change them. The user can establish a connection between the programming device and PLC and casn view variables online.
2	Reset SFC	The same functionality as above and also: Animation panel can be use for control (e.g. disable steps, disable transitions, force steps, etc.).
3	Change data	The same functionality as above and also: The user can change literals online.
4	Force data	The same functionality as above and also: Forcing variables.
5	Download	The same functionality as above and also: The user can download the program to the PLC. **Note:** To download the configuration, you at least need the access rights **Change configuration**.
6	Change program	The same functionality as above and also: The user can make any changes to the program, but not to DFBs or EFBs.
7	Change configuration	The same functionality as above and also: The user can change the PLC configuration.
8	Tools	The same functionality as above and also: The user can use Concept DFB, Concept EFB and Concept Converter.
9	Supervisor	The same functionality as above and also: The user can use Concept Security in Supervisor mode (set up users, activate/deactivate Concept Security).

Access Rights for the Main Menu File

The following table shows the minimum access rights required in Concept for the menu commands in the main menu **File**:

Menu commands in the main menu File	Minimum access rights needed
New Project	Change program
Open / **Close**	Read only
Open / **Close** (replacing/deleting EFBs/DFBs; error messages: FFB does not exist; FFB formula parameter was changed, DFB was changed internally)	Change program
Save project	Change data
Save project as....	Change data
Optimize project...	Change program
New section...	Change program
Open section...	Read only
Delete section...	Change program
Section properties... (read)	Read only
Section properties... (write)	Change program
Section Memory	Read only
Import...	Change program
Export...	Read only
Print...	Read only
Printer setup...	Read only
Exit	Read only

Access Rights for the Main Menu Edit

The following table shows the minimum access rights required in Concept for the menu commands in the main menu **Edit**:

Menu commands in the main menu Edit	Minimum access rights needed
Undo: Delete	Change program
Cut	Change program
Copy	Read only
Insert	Change program
Delete	Change program
Select all	Read only
Deselect all	Read only
Goto line... (text languages)	Read only
Goto counterpart (text languages)	Read only
Expand statement (text languages)	Change program
Lookup variables (text languages)	Change program
Search... (text languages)	Read only
Find Next (text languages)	Read only
Replace... (text languages)	Change program
Insert text file... (text languages)	Change program
Save as text file... (text languages)	Read only
Open Column (LL984 Editor)	Read only
Open Row (LL984 Editor)	Read only
Close Column (LL984 Editor)	Read only
Close Row (LL984 Editor)	Read only
DX Zoom... (LL984 Editor)	Read only
Reference Zoom (LL984 Editor)	Read only
Offset References... (LL984 Editor)	Read only
Replace References (LL984 Editor)	Read only

Access Rights for the Main Menu View

The following table shows the minimum access rights required in Concept for the menu commands in the main menu **View** (only for FBD, LD and SFC):

Menu commands in the main menu View	Minimum access rights needed
Overview	Read only
Normal	Read only
Expanded	Read only
Zoom in	Read only
Zoom out	Read only
Grid	Read only
Page breaks	Read only

Access Rights for the Main Menu Objects

The following table shows the minimum access rights required in Concept for the menu commands in the main menu **Objects**:

Menu commands in the main menu Objects	Minimum access rights needed
Properties (read) (only for FBD, LD and SFC)	Read only
Properties (write) (only for FBD, LD and SFC)	Change program
Select	Read only
Text	Change program
Replace variables...	Change program
Link	Change program
Vertical Link (LD Editor)	Change program
FFB: Last Type (FBD, LD Editor)	Change program
Invert input/output (FBD, LD Editor)	Change program
Insert Macro... (FBD Editor)	Change program
FFB selection... (FBD, LD Editor)	Change program
Replace FFBs... (FBD, LD Editor)	Change program
FFB Show execution order (FBD Editor)	Read only
Reverse FFB execution order (FBD Editor)	Change program
Insert contacts, coils (LD Editor)	Change program
Select column structure (SFC Editor)	Change program
Select row structure (SFC Editor)	Change program
Insert steps, transitions (SFC Editor)	Change program
Insert FFB, **Download**, **Save** etc. (IL Editor)	Change program
Insert FFB, **Assignment**, **Operators**, **Declaration** etc. (ST Editor)	Change program
Coils, **Insert contacts** (LL984 Editor)	Change program

Access Rights for the Main Menu Project

The following table shows the minimum access rights required in Concept for the menu commands in the main menu **Project**:

Menu commands in the main menu Project	Minimum access rights needed
Properties (write)	Change program
Memory Prediction	Read only
PLC configuration	Change configuration
Project Browser (write)	Change program
Execution order... (write)	Change program
Variable declarations... (write)	Change program
ASCII Messages	Read only
Search...	Read only
Trace	Read only
Find Next	Read only
Search Results	Read only
Used references...	Read only
Analyze section	Read only
Analyze program	Read only
Synchronize versions of nested DFBs	Read only
Code generation options...	Supervisor

Access Rights for the Main Menu Online

The following table shows the minimum access rights required in Concept for the menu commands in the main menu **Online**:

Menu commands in the main menu Online	Minimum access rights needed
Connect... (view only)	Read only
Connect... (change data)	Reset SFC
Connect... (change program)	Download
Connect... (change configuration)	Download
Disconnect...	Read only
Online control panel... (all commands)	Download
Single sweep trigger	Download
Controller status......	Read only
Online events...	Read only
Online diagnostics (read)	Read only
Online diagnostics (acknowledge entries)	Change data
Record changes	Change program
Object information...	Read only
Memory statistics...	Read only
Download... (IEC Program, 984 Ladder Logic, ASCII Messages, Status-RAM, Extended Memory)	Download
Download... (configuration)	Change configuration
Download changes...	Change program
Upload... (Status-RAM, Extended Memory)	Change data
Upload... (IEC Program, 984 Ladder Logic, ASCII Messages, Status-RAM)	Change program
Upload... (configuration)	Change configuration
Reference Data Editor (read only)	Read only
Reference Data Editor (write)	Change data
Reference Data Editor (force)	Force data
Disabled discretes...	Change data
Activate animation	Read only
Change literals during animation	Change data
Animation Panel (SFC Editor)	SFC Animation Panel
Animation Panel (forcing SFC steps)	SFC Animation Panel
Animation Panel (Resetting an SFC string)	SFC Animation Panel
Save animation (IL, ST Editor)	Read only

Menu commands in the main menu Online	Minimum access rights needed
Restore animation (IL, ST Editor)	Read only
Direct-mode 984LL Editor... (LL984 Editor)	Read only
power flow (LL984 Editor)	Read only
Power flow with contact state (LL984 Editor)	Read only
Trace (LL984 Editor)	Read only
ReTrace (LL984 Editor)	Read only

Access Rights for the Main Menu Options

The following table shows the minimum access rights required in Concept for the menu commands in the main menu **Options**:

Menu commands in the main menu Options	Minimum access rights needed
Confirmations...	Change program
Preferences → **Common...**	Change program
Preferences → **Graphical Editor...**	Change program
Preferences → **Analysis...**	Change program
Preferences → **IEC Extensions...**	Change program
Save settings	Change program
Save settings on close	Change program

Access Rights for the Main Menu Window

The following table shows the minimum access rights required in Concept for the menu commands in the main menu **Window**:

Menu commands in the main menu Window	Minimum access rights needed
Cascade	Read only
Slit window	Read only
Tile	Read only
Arrange icons	Read only
Close all	Read only
Messages	Read only
Name of Open Sections	Read only

Changing Passwords

Introduction

This section describes the steps necessary to change the system administrator's password and enter new users.

Changing the System Administrator's password

The following steps are only necessary when you start Concept Security for the first time after installing Concept. They describe the procedure for changing the system administrator's password:

Step	Action
1	Start access management by double clicking on the Concept Security icon.
2	Enter the user name for the **`supervisor`** and confirm with **OK**. Entering a password is not necessary in this case.
3	Press the **Change Password** command button.
4	Enter a password in the **Password** text box. **Note:** The password is context sensitive.
5	To confirm the password, enter the same password in the **Confirm New Password** text box. **Reaction:** If the two entries are identical, the command button **OK** is enabled.
6	Confirm the change by pressing the **OK** button
7	Exit access management with the command button **Exit**.

Entry for a user and the access rights

To enter users, assign access rights and activate Concept Security, follow these steps:

Step	Action
1	Start access management by double clicking on the Concept Security icon.
2	Enter a user name with supervisor access rights, enter the password and confirm with **OK**.
3	Select the **User** tab.
4	Press the **Add** command button.
5	Enter the user name (at least 2, maximum 16 characters) and confirm with **OK**.
6	In the **Access Rights:** list box, select the desired access rights and confirm with the command button **OK**.
7	Exit access management with the command button **Exit**.
8	To change the password for the new user, follow the procedure Changing the System Administrator's password. Enter the user name for the user that was just defined.

Activating Access Rights

Activating access rights

To activate access rights, follow these steps:

Step	Action
1	Start access management by double clicking on the Concept Security symbol.
2	Enter a user name with supervisor access rights, enter the password and acknowledge with **OK**.
3	Select the register **Options**.
4	Activate the check box **Password Required**.
5	Exit access management with the command button **Exit**. **Reaction:** Concept, Concept DFB, Concept EFB, etc. can only be started by authorized users and with the access rights defined for them.

Protecting Projects/DFBs

Introduction

With Concept Security, you can protect projects/DFBs from being changed. Protected projects can only be loaded on the PLC but cannot be changed. Protected DFBs can only be used and cannot be changed.

Protecting projects/DFBs

To protect projects/DFBs, follow these steps:

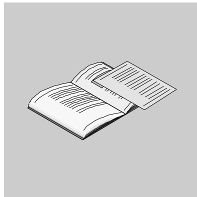
Step	Action
1	Start access management by double clicking on the Concept Security symbol.
2	Enter a user name with supervisor access rights, enter the password and confirm with **OK**.
3	Select the **Protect** register.
4	Press the command button **Select** and select the project/DFB to be protected. Confirm with **OK**. **Reaction:** The selected project/DFB will appear in a list box.
5	Select the project/DFB in the list box and press **Protect**. **Reaction:** The dialog box **Enter Password** is opened.
6	Enter a password for **Password** and acknowledge it by entering the same password for **Confirm Password**. Press **OK**. **Reaction:** The project/DFB is now protected. This is identified by a **(c)** in the list box.
7	In order to locate protected projects/DFBs quickly, it is advisable to save the list in the **Program/DFB** list box using **Save list...** .

Deactivate protection for projects/DFBs

To deactivate protection for projects/DFBs, follow these steps:

Step	Action
1	Start access management by double clicking on the Concept Security symbol.
2	Enter a user name with supervisor access rights, enter the password and confirm with **OK**.
3	Select the **Protect** register.
4	Press the command button **Select** and select the protected project/DFB that should have protection deactivated. Confirm with **OK**. **Reaction:** The selected project/DFB will appear in a list box. **or** Use **Load list...** to load a previously saved list. **Reaction:** All projects/DFBs in the loaded list will appear in the list box.
5	Select the project/DFB from the list box (identified by **(c)**) and press **Unprotect**. **Reaction:** The **Enter Password** dialog box is opened.
6	Enter the password for **Password** and press **OK**. **Reaction:** The project/DFB is no longer protected. This is identified by the **(c)** not being shown in the list box.

Appendices



At a Glance

Overview

Additional information that is not necessarily required for an understanding of the documentation.

What's in this Appendix?

The appendix contains the following chapters:

Tables of PLC-dependent Performance Attributes

A

Introduction

Overview

The performance attributes of the different hardware platforms (Quantum, Compact, Momentum and Atrium) can be found in the following tables.

What's in this Chapter?

This chapter contains the following topics:

Performance of Quantum

IEC and LL984 Support

Availability of IEC and LL984 support:

Performance	CPU type					
	113 02	**113 03**	**213 04**	**424 0x**	**434 12**	**534 14**
LL984 only	x	-	-	-	-	-
IEC only (Stripped Exec)	x	x	x	-	-	-
IEC and LL984	-	x	x	x	x	x
x = available - = not available						

Special Performance Attributes

Availability of special performance attributes:

Performance	CPU type					
	113 02	**113 03**	**213 04**	**424 0x**	**434 12**	**534 14**
LL984 Hot Standby	x	x	x	x	x	x
IEC Hot Standby	-	-	-	-	x	x
Interrupt processing with HLI (LL984 only)	x	x	x	x	x	x
Split memory (LL984 only with separate software	-	-	-	-	-	-
Support for XMIT loadable (LL984 only)	x	x	x	x	x	x
Support for XMIT EFB (IEC only)	-	-	-	-	-	-
Support for XXMIT EFB (IEC only)	x	x	x	x	x	x
Upload of the user program	x	x	x	x	x	x
Support of the Modbus function codes 42 (IEC only)	x	x	x	x	x	x
Password protection of connection structure with PLC	-	-	-	-	-	-
PCMCIA support	-	-	-	-	-	-
Flash memory for program and configuration	-	-	-	-	x	x
Remote Terminal Unit (RTU) configuration extension	-	-	-	-	-	-
Profibus DP configuration extension	x	x	x	x	x	x
Cyclical data exchange for configuration extension	x	x	x	x	x	x
Code generation options: Include diagnosis information	x	x	x	x	x	x

Performance	CPU type					
	113 02	113 03	213 04	424 0x	434 12	534 14
Code generation options: Fastest code	-	-	x	x	x	x
MMS Ethernet configuration extension	x	x	x	x	x	x
ASCII Messages	x	x	x	x	x	x
Peer Cop	x	x	x	x	x	x
RIO (Remote I/O)	x	x	x	x	x	x
DIO (Distributed I/O)	x	x	x	x	x	x
SYMAX I/O	x	x	x	x	x	x
800 I/O	x	x	x	x	x	x
LonWorks	x	x	x	x	x	x
A120 I/O	-	-	-	-	-	-
x = available - = not available						

Buses

Availability of the buses:

Performance	CPU type					
	113 02	**113 03**	**213 04**	**424 0x**	**434 12**	**534 14**
Modbus	x	x	x	x	x	x
Modbus Plus	x	x	x	x	x	x
Ethernet (TCP/IP)	x	x	x	x	x	x
Ethernet (SY/MAX)	x	x	x	x	x	x
Interbus	x	x	x	x	x	x
Interbus: PCP loadable (LL984 only)	x	x	x	x	x	x
Interbus: PCP-EFB (IEC only)	x	x	x	x	-	-
INTERBUS G4 (Generic Bus)	-	x	x	-	x	x
LonWorks (Echelon)	using NOL 911 xx and LL984	using NOL 911 xx and LL984	using NOL 911 xx and LL984	using NOL 911 xx and LL984	using NOL 911 xx and LL984	using NOL 911 xx and LL984
MVB (MultiVehicleBus)	-	-	-	-	-	-
x = available - = not available						

Block Libraries

Availability of the block libraries:

Performance	CPU type					
	113 02	**113 03**	**213 04**	**424 0x**	**434 12**	**534 14**
AKFEFB (IEC only)	x	x	x	x	x	x
ANA_IO (IEC only)	x	x	x	x	x	x
COMM (IEC only)	x	x	x	x	x	x
CONT_CTL (IEC only)	x	x	x	x	x	x
DIAGNO (IEC only)	x	x	x	x	x	x
EXPERTS (IEC only)	x	x	x	x	x	x
EXTENDED (IEC only)	x	x	x	x	x	x
FUZZY (IEC only)	x	x	x	x	x	x
HANDTABLEAU (IEC only)	x	x	x	x	x	x
IEC (IEC only	x	x	x	x	x	x
LIB984 (IEC only)	x	x	x	x	x	x
SYSTEM (IEC only)	x	x	x	x	x	x
LL984 (LL984 only)	x	x	x	x	x	x
x = available - = not available						

Utilities

Availability of utilities:

Performance	CPU type					
	113 02	**113 03**	**213 04**	**424 0x**	**434 12**	**534 14**
Concept DFB	x	x	x	x	x	x
Concept EFB	x	x	x	x	x	x
Concept SIM	x	x	x	x	x	x
Concept PLCSIM32	x	x	x	x	x	x
Concept security	x	x	x	x	x	x
Concept EXECLoader	x	x	x	x	x	x
Concept-Converter	x	x	x	x	x	x
Modsoft converter	x	x	x	x	x	x
ModConnect tool	x	x	x	x	x	x
x = available - = not available						

Runtime System

Runtime System

Performance	CPU type					
	113 02	**113 03**	**213 04**	**424 0x**	**434 12**	**534 14**
16 bit CPU	x	x	x	x	-	-
32 bit CPU	-	-	-	-	x	x
x = available - = not available						

Available Memory for User Program

Available memory for user program

Performance	CPU type					
	113 02	**113 03**	**213 04**	**424 0x**	**434 12**	**534 14**
IEC only runtime system	125k	375k	612k	-	-	-
IEC and LL984 runtime system	-	160k	330k	460k	800k	2500k
LL984 only runtime system	-	-	-	-	-	-
x = available - = not available						

Different Performance Attributes

Availability of different performance attributes:

Performance	CPU type					
	113 02	**113 03**	**213 04**	**424 0x**	**534 14**	**534 14**
Battery adapter required for backing up IEC programs	-	-	-	-	-	-
Floating point processor	-	-	x	x	x	x
Floating point emulation (IEC)	x	x	-	-	-	-
x = available - = not available						

Performance Attributes of Compact

IEC and LL984 Support

Availability of IEC and LL984 support:

Performance	CPU type			
	258 (512k)	265 (512k)	275 (512k)	285 (1M)
LL984 only	-	-	-	-
IEC only (Stripped Exec)	-	-	-	-
IEC and LL984	x	x	x	x
x = available - = not available				

Special Performance Attributes

Availability of special performance attributes:

Performance	CPU type			
	258 (512k)	265 (512k)	275 (512k)	285 (1M)
LL984 Hot Standby	-	-	-	-
IEC Hot Standby	-	-	-	-
Interrupt processing with HLI (LL984 only)	-	-	-	-
Split memory (LL984 only with separate software	x	x	x	x
Support for XMIT loadable (LL984 only)	x	x	x	x
Support for XMIT EFB (IEC only)	-	-	-	-
Support for XXMIT EFB (IEC only)	x	x	x	x
Upload of the user program	x	x	x	x
Support of Modbus function code 42 (IEC only)	x	x	x	x
Password protection of connection structure with PLC	x	x	x	x
PCMCIA support	-	-	x	x
Flash memory for program and configuration	x	x	x	x
Remote Terminal Unit (RTU) configuration extension	x	x	x	x

Performance	CPU type			
	258 (512k)	**265 (512k)**	**275 (512k)**	**285 (1M)**
Profibus DP configuration extension	-	-	-	-
Cyclical data exchange for configuration extension	-	-	-	-
Code generation options: Include diagnosis information	x	x	x	x
Code generation options: Fastest code	x	x	x	x
MMS Ethernet configuration extension	-	-	-	-
ASCII Messages	-	-	-	-
Peer Cop	-	x	x	x
RIO (Remote I/O)	-	-	-	-
DIO (Distributed I/O)	-	-	-	-
SYMAX I/O	-	-	-	-
800 I/O	-	-	-	-
LonWorks	-	-	-	-
A120 I/O	x	x	x	x
x = available - = not available				

Buses

Availability of the buses:

Performance	CPU type			
	258 (512k)	265 (512k)	275 (512k)	285 (1M)
Modbus	x	x	x	x
Modbus Plus	using BridgeModule	x	x	x
Ethernet (TCP/IP)	using Bridge Module	using Bridge Module	using Bridge Module	using Bridge Module
Ethernet (SY/MAX)	-	-	-	-
Interbus	using BKF xxx	using BKF xxx	using BKF xxx	using BKF xxx
Interbus: PCP loadable (LL984 only)	-	-	-	-
Interbus: PCP-EFB (IEC only)	-	-	-	-
LonWorks (Echelon)	-	-	-	-
MVB (MultiVehicleBus)	x	x	x	x
x = available - = not available				

Block Libraries

Availability of block libraries:

Performance	CPU type			
	258 (512k)	265 (512k)	275 (512k)	285 (1M)
AKFEFB (IEC only)	x	x	x	x
ANA_IO (IEC only)	x	x	x	x
COMM (IEC only)	-	x	x	x
CONT_CTL (IEC only)	x	x	x	x
DIAGNO (IEC only)	x	x	x	x
EXPERTS (IEC only)	x	x	x	x
EXTENDED (IEC only)	x	x	x	x
FUZZY (IEC only)	x	x	x	x
HANDTABLEAU (IEC only)	x	x	x	x
IEC (IEC only)	x	x	x	x
LIB984 (IEC only)	x	x	x	x
SYSTEM (IEC only)	x	x	x	x
LL984 (LL984 only)	x	x	x	x
x = available - = not available				

Utilities

Availability of utilities:

Performance	CPU type			
	258 (512k)	265 (512k)	275 (512k)	285 (1M)
Concept DFB	x	x	x	x
Concept EFB	x	x	x	x
Concept SIM	x	x	x	x
Concept PLCSIM32	x	x	x	x
Concept Security	x	x	x	x
Concept EXECLoader	x	x	x	x
Concept-Converter	x	x	x	x
Modsoft converter	x	x	x	x
Concept-ModConnect	-	-	-	-
x = available - = not available				

Runtime System

Runtime system

Performance	CPU type			
	258 (512k)	265 (512k)	275 (512k)	285 (1M)
16 bit CPU	-	-	-	-
32 bit CPU	x	x	x	x
x = available - = not available				

Different Performance Attributes

Availability of different performance attributes:

Performance	CPU type			
	258 (512k)	265 (512k)	275 (512k)	285 (1M)
Battery adapter required for backing up IEC programs	-	-	-	-
Floating point processing	-	-	-	-
Floating point emulation	x	x	x	x
x = available - = not available				

Performance Attributes of Momentum

IEC and LL984 Support

Availability of IEC and LL984 support:

Performance	CPU type				
	700 00 700 10 780 00	760 00	760 10 780 10	960 20 980 20	960 30 980 30
LL984 only	x	x	x	x	x
IEC only	-	x	x	-	x
IEC and LL984	-	-	-	-	-
x = available - = not available					

Special Performance Attributes

Availability of special performance attributes:

Performance	CPU type				
	700 00 700 10 780 00	760 00	760 10 780 10	960 20 980 20	960 30 980 30
LL984 Hot Standby	-	-	-	-	-
IEC Hot Standby	-	-	-	-	-
Interrupt processing with HLI (LL984 only)	-	-	-	-	-
Split memory (LL984 only with separate software	-	-	-	-	-
Support for the XMIT blocks (LL984 only)	x	x	x	x	x
Support for XMIT EFB (IEC only)	-	-	-	-	-
Support for XXMIT EFB (IEC only)	x	x	x	x	x
Upload of the user program	x	x	x	x	x
Support of Modbus function code 42 (IEC only)	-	x	x	-	x
Password protection of connection structure with PLC	-	-	-	x	x
PCMCIA support	-	-	-	-	-

Performance	CPU type				
	700 00 700 10 780 00	**760 00**	**760 10 780 10**	**960 20 980 20**	**960 30 980 30**
Flash memory for program and configuration (LL984)	x	x	x	x	x
Flash memory for program and configuration (IEC)	-	-	x	-	x
Remote Terminal Unit (RTU) configuration extension	-	-	-	-	-
Profibus DP configuration extension	-	-	-	-	-
Cyclical data exchange for configuration extension	-	-	-	-	-
Code generation options: Include diagnosis information	-	-	-	-	-
Code generation options: Fastest code	-	-	-	-	-
MMS Ethernet configuration extension	-	-	-	-	-
ASCII Messages	-	-	-	-	-
Peer Cop	x	x	x	x	x
RIO (Remote I/O)	-	-	-	-	-
DIO (Distributed I/O)	-	-	-	-	-
TIO (Terminal I/O	x	x	x	x	x
SYMAX I/O	-	-	-	-	-
800 I/O	-	-	-	-	-
LonWorks	-	-	-	-	-
A120 I/O	-	-	-	-	-
x = available - = not available					

Buses

Availability of the buses:

Performance	CPU type				
	700 00 700 10 780 00	**760 00**	**760 10 780 10**	**960 20 980 20**	**960 30 980 30**
Modbus (with ring card)	x	x	x	x	x
Modbus Plus (with ring card)	x	x	x	x	x
Ethernet (TCP/IP)	-	-	-	x (LL984 only)	x
Ethernet (SY/MAX)	-	-	-	-	-
Interbus	x	x	x	x	x
Interbus: PCP loadable (LL984 only)	-	-	-	-	-
Interbus: PCP-EFB (IEC only)	-	-	-	-	-
LonWorks (Echelon)	-	-	-	-	-
MVB (MultiVehicleBus)	-	-	-	-	-
x = available - = not available					

Block Libraries

Availability of the block libraries:

Performance	CPU type				
	700 00 700 10 780 00	**760 00**	**760 10 780 10**	**960 20 980 20**	**960 30 980 30**
AKFEFB (IEC only)	-	x	x	-	x
ANA_IO (IEC only)	-	x	x	-	x
COMM (IEC only)	-	-	-	-	x
CONT_CTL (IEC only)	-	x	x	-	x
DIAGNO (IEC only)	-	x	x	-	x
EXPERTS (IEC only)	-	-	-	-	x
EXTENDED (IEC only)	-	x	x	-	x
FUZZY (IEC only)	-	x	x	-	x
HANDTABLEAU (IEC only)	-	-	-	-	x
IEC (IEC only)	-	x	x	-	x
LIB984 (IEC only)	-	x	x	-	x
SYSTEM (IEC only)	-	x	x	-	x
LL984 (LL984 only)	x	x	x	x	x
x = available - = not available					

Utilities

Availability of utilities:

Performance	CPU type				
	700 00 700 10 780 00	**760 00**	**760 10 780 10**	**960 20 980 20**	**960 30 980 30**
Concept DFB	-	x	x	-	x
Concept EFB	-	x	x	-	x
Concept SIM	-	x	x	-	x
Concept PLCSIM32	-	x	x	-	x
Concept security	-	x	x	-	x
Concept EXECLoader	x	x	x	x	x
Concept-Converter	x	x	x	x	x
Modsoft converter	x	x	x	x	x
Concept-ModConnect	x	x	x	x	x
x = available - = not available					

Runtime System

Runtime System

Performance	CPU type				
	700 00 700 10 780 00	**760 00**	**760 10 780 10**	**960 20 980 20**	**960 30 980 30**
16 bit CPU	x	x	x	x	x
32 bit CPU	-	-	-	-	-
x = available - = not available					

Different Performance Attributes

Availability of different performance attributes:

Performance	CPU type				
	700 00 700 10 780 00	**760 00**	**760 10 780 10**	**960 20 980 20**	**960 30 980 30**
Battery adapter required for backing up IEC programs	-	x	-	-	-
Floating point processor	-	-	-	-	-
Floating point emulation (IEC)	-	x	x	-	x
x = available - = not available					

Performance Attributes of Atrium

IEC and LL984 Support

Availability of IEC and LL984 support:

Performance	CPU type 121 01 (2M) 241 01 (4M) 241 11 (4M)
LL984 only	-
IEC only (Stripped Exec)	x
IEC and LL984	-
x = available - = not available	

Special Performance Attributes

Availability of special performance attributes:

Performance	CPU type 121 01 (2M) 241 01 (4M) 241 11 (4M)
LL984 Hot Standby	-
IEC Hot Standby	-
Interrupt processing with HLI (LL984 only)	-
Split memory (LL984 only with separate software	-
Support for XMIT loadable (LL984 only)	-
Support for XMIT EFB (IEC only)	-
Support for XXMIT EFB (IEC only)	-
Upload of the user program	x
Support of Modbus function code 42 (IEC only)	x
Password protection of connection structure with PLC	-
PCMCIA support	-

Performance	CPU type
	121 01 (2M) **241 01 (4M)** **241 11 (4M)**
Flash memory for program and configuration	-
Remote Terminal Unit (RTU) configuration extension	-
Profibus DP configuration extension	-
Cyclical data exchange for configuration extension	-
Code generation options: Include diagnosis information	-
Code generation options: Fastest code	-
MMS Ethernet configuration extension	-
ASCII Messages	-
Peer Cop	x
RIO (Remote I/O)	-
DIO (Distributed I/O)	-
SYMAX I/O	-
800 I/O	-
LonWorks	-
A120 I/O	-
x = available - = not available	

Buses

Availability of the buses:

Performance	CPU type 121 01 (2M) 241 01 (2M) 241 11 (4M)
Modbus	-
Modbus Plus	x
Ethernet (TCP/IP)	-
Ethernet (SY/MAX)	-
Interbus	x x x
Interbus: PCP loadable (LL984 only)	-
Interbus: PCP-EFB (IEC only)	-
Profibus	- - -
LonWorks (Echelon)	-
MVB (MultiVehicleBus)	-
x = available - = not available	

Block Libraries

Availability of block libraries:

Performance	CPU type 121 01 (2M) 241 01 (2M) 241 11 (4M)
AKFEFB (IEC only)	x
ANA_IO (IEC only)	x
COMM (IEC only)	x
CONT_CTL (IEC only)	x
DIAGNO (IEC only)	x
EXPERTS (IEC only)	x
EXTENDED (IEC only)	x
FUZZY (IEC only)	x
HANDTABLEAU (IEC only)	x
IEC (IEC only)	x
LIB984 (IEC only)	x
SYSTEM (IEC only)	x
LL984 (LL984 only)	-
x = available - = not available	

Utilities

Availability of utilities:

Performance	CPU type 121 01 (2M) 241 01 (2M) 241 11 (4M)
Concept DFB	x
Concept EFB	x
Concept SIM	x
Concept PLCSIM32	x
Concept Security	x
Concept EXECLoader	x
Concept-Converter	x
Modsoft converter	x
Concept-ModConnect	-
x = available - = not available	

Runtime System

Runtime system

Performance	CPU type 121 01 (2M) 241 01 (2M) 241 11 (4M)
16 bit CPU	-
32 bit CPU	x
x = available - = not available	

Different Performance Attributes

Availability of different performance attributes:

Performance	CPU type **121 01 (2M)** **241 01 (2M)** **241 11 (4M)**
Battery adapter required for backing up IEC programs	-
Floating point processor	- x x
Floating point emulation	x - -
x = available - = not available	

Windows interface

B

At a Glance

Overview

The chapter describes the most important properties of Concept's Windows interface. Further information can be found in the Microsoft Windows manuals.

What's in this Chapter?

This chapter contains the following sections:

B.1 Window

At a Glance

Overview

This section describes the types of windows and window elements in Windows.

What's in this Section?

This section contains the following topics:

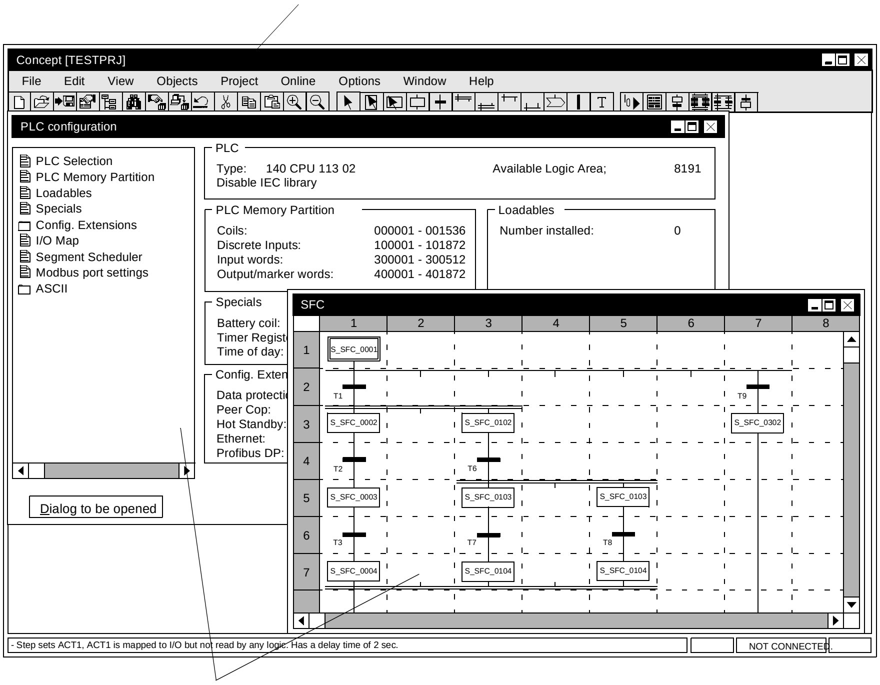
Window Types

Introduction

In Windows there are two types of windows:
- Application Window
- Document Window

Types of window:

Application window (project)



Document window (PLC configuration, section)

Application Window

When Concept is started the application window is opened on your desktop. The application window can be moved to any position on the desktop. Alternatively it can be minimized to a button on the task bar.

A project can be opened or created in this application window. The name of the project then appears in the title bar of the application window.

Document Window

After opening or creating a project you can open different document windows. Document windows are, for example, sections in which a user program is created or the document window of the PLC configuration.

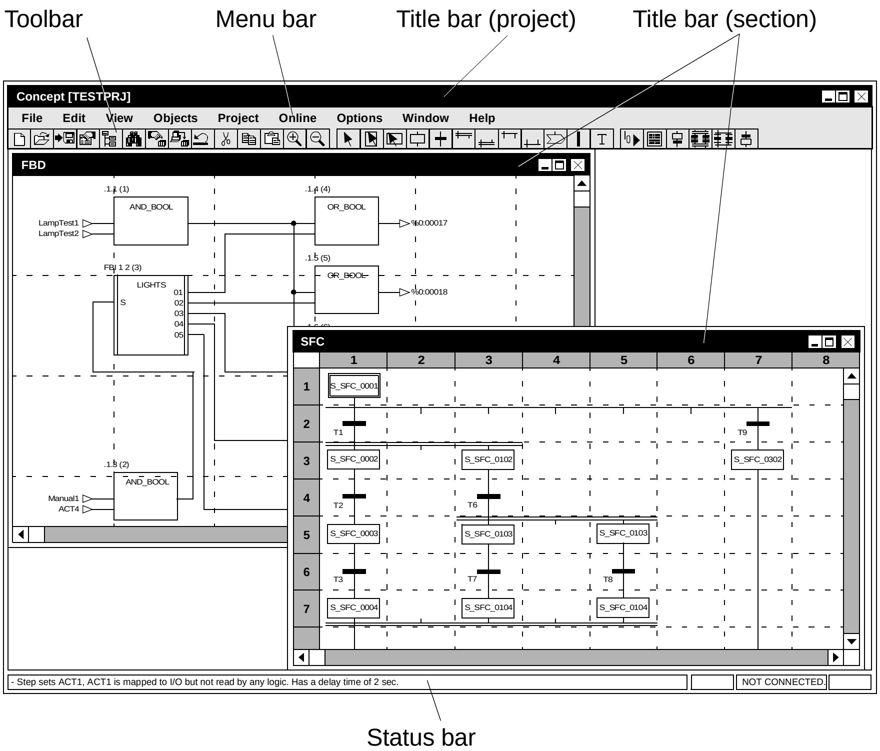
Several document windows can be open simultaneously, but only one of these can be active. An active document window can be recognized by the color of the title bar.

Depending on the active document window the menu commands change in the pull down menus and the tool bar of the application window.

Elements of a window

At a Glance

This section describes the Concept specific elements of a window.
Elements of a window:



Title bar

A project's title bar shows the name of the active application (i.e. Concept) and the name of the project. When coupled with a PLC the node address of the PLC is indicated in angled brackets (<>). If this PLC is on another network the routing path is also indicated.

If a document window (e.g. a section) is enlarged to full screen, i.e. the section takes up the entire application window, the name of the document window (e.g. the section name) appears in the title bar.

Document windows which are not enlarged to full screen have their own title bar in which the name of the document window is indicated.

Menu Bar

The menu bar of the application window contains various main menus. The contents of the menu bar depend on the active document window.

Toolbar

The toolbar consists of buttons which correspond to a menu command on the pull-down menus. The range and content of the toolbar depend on which window is active.

There are three different ways a button can be represented:
- grayed
  The command is currently unavailable. One or more other commands must be executed before the desired button can be used.
- unpressed
  The command can be selected.
- pressed
  The command is active.

Status bar

The appearance of the status bar depends on whether the project is open and the programming language used in the section.

In the first part of the status bar various information is displayed depending on the selected object.

- If a dialog box is open or a menu command or button has been selected some help will be given about it. To display the help select a menu command or a button with the left mouse button and hold it down. A short description of the menu command or button appears in the status bar. To execute the menu command/ button release the mouse button. If execution of the menu command/button is not required, move the pointer away from the active area (the description in the status bar disappears) and then release the mouse button.
- If an FFB, a parameter to an input/output, a step or a transition has been selected, a comment about the selected object is displayed. With parameters and transitions the assigned direct address (only in case of located variables) is also displayed.

The second part of the status bar (status of the active section) indicates whether the section is in animation mode or the section is disabled.

- ANIMATED
  The section is animated.
- INHIBITED
  The section is inhibited and will not be processed.

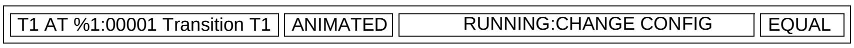
The third part of the status bar indicates the status of the PLC.

- NOT CONNECTED.
  The programming device is not coupled with a PLC.
- STOPPED
  The program on the PLC is suspended.
- RUNNING: CHANGE CONFIG
  The program on the PLC is running and was connected with the access **Change Configuration**.

In the fourth part of the status bar the program status between the PLC and programming device is displayed. This display only appears if a project is open and the programming device with PLC is online.

- EQUAL
  The program on the programming device and the PLC is consistent.
- UNEQUAL
  The program on the programming device and the PLC is not consistent. To establish consistency use the menu command **Online** → **Download...** .
- MODIFIED
  The program on the programming device was modified. The modifications can be made online in the PLC with the menu command **Online** → **Download changes**.

Status bar:



B.2 Menu commands

Menu commands

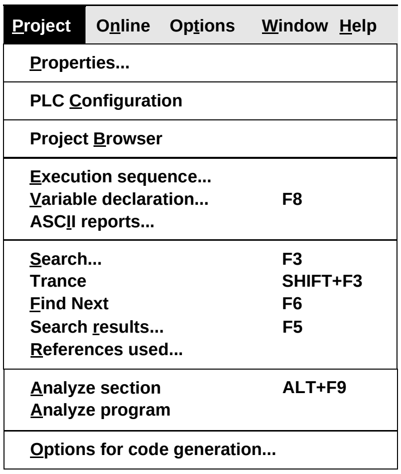
At a Glance

The titles of the individual menus are displayed in the menu bar. The menu commands are listed in the pull-down menus. As in Windows, each Concept window and dialog box has a system menu. This menu is opened using the small box in the top left-hand corner of the window.

A pull-down menu is opened by left-clicking on the title of the menu. To go directly to a menu command, drag the mouse pointer down the menu and then release the mouse button.

The menu can be closed by clicking on the title of the menu or anywhere outside of the menu.

Typical pull-down menu:



Underlined letter

A main menu (menu title) and subsequently a menu command can be selected by holding down **Alt** and simultaneously entering the underlined letter in the menu title and then that of the menu command. If, for instance, from the menu **Project** you want to execute the menu command **Search...** press **Alt**+**P** to open the menu and then **Alt**+**S** to execute the menu command.

Grayed out menu command

The command is currently unavailable. One or more other commands must be executed before the desired menu command can be executed.

Suspension points (…) after the menu command	On execution of this menu command a dialog box appears with options, which must be selected before execution.
Check mark (√) before the menu command	The menu command is active. If the menu command is selected the check mark disappears and the menu command is inactive. The check mark is mostly used to identify active modes (e.g. normal display, dial in mode etc.).
Shortcut keys	The key combinations (e.g. **F8**, **Alt**+**F9**, **Ctrl**+**R**) after the menu command are shortcut keys for executing this menu command. Using this key or key combination the menu command can be selected, without having to open the menu.

B.3 Dialog boxes

Dialog boxes

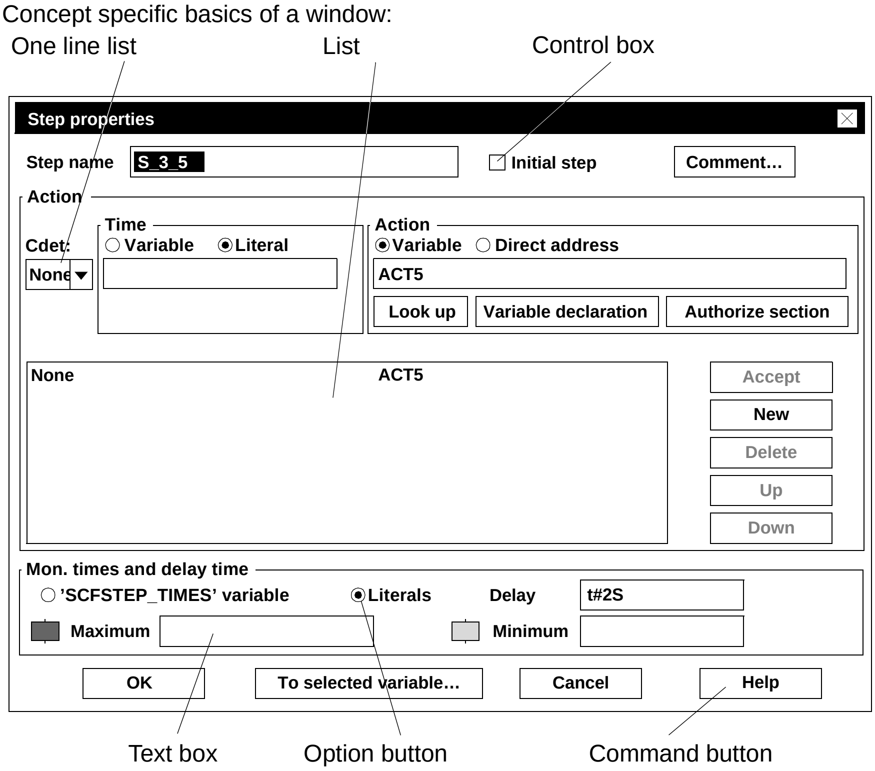
At a Glance

In Concept dialog boxes are displayed if additional information is required from you in order to perform a particular task. Potentially necessary information is also communicated in this way.

Most dialog boxes contain options which can be selected, textboxes, in which text can be entered, and buttons which can be pressed.

Grayed out options are currently not available. One or more other commands must be executed, or options selected or deselected, before the desired option can be activated.

Concept specific basics of a window:



Command buttons

Command buttons are used to initiate actions immediately, e.g. executing or aborting a command. Command buttons include e.g. **OK**, **Abort** and **Help**.

Command buttons followed by suspension points (…), open a further dialog box. A command button with a "greater than" sign (>>) extends the active dialog box.

The standard setting is identified by a dark margin. This command button can be selected by pressing **Enter**.

To close a dialog box without executing a command select the command button **Cancel**.

Text boxes

Information (text) is entered into a text box.

If you enter an empty text box an insertion point appears in the far left of the box. The entered text begins at this insertion point. If text is already present within the respective box, it will be selected and replaced by the new text automatically. The text can, however, also be deleted by pressing **Delete** or **Backspace**.

Lists

In a list the available selection possibilities are listed. If more possibilities are available than fit into the list, the scrollbar or the arrow keys can be used to move within the list.

As a rule only a single entry can be chosen form the list. There are, however, some cases in which several entries can be chosen, e.g. when opening sections.

One line lists

A single line list box initially appears as a rectangular box, in which the current selection (the default value) is selected. If the arrow in the right of the box is selected, a list of the available selection possibilities opens. If more possibilities are available than fit into the list, then the scrollbar or arrow keys can be used to move around the list.

Option buttons

Option buttons represent mutually exclusive options. In each case only one option can be chosen.

The selected option button is identified by a black dot.

If the option name contains an underlined letter, the option button can be activated from any position in the dialog box by holding down **Alt** and entering the underlined letter.

Check box

A check box next to an option means that the option can be activated or deactivated. Any number of check box options can be activated.

Activated options are identified by an X or a check mark (√).

If the option name contains an underlined letter, the check box can be activated or deactivated from any position in the dialog box by holding down **Alt** and entering the underlined letter.

B.4 Generating a project symbol

Creating a Project Symbol in a Program Group

Introduction

Creating a project symbol allows you to immediately load a certain project and/or connect to a PLC when opening Concept. In this way, one or more program groups can be created, which e.g. contain all the projects in a system.

Note: A symbol can only be created for an existing project. Otherwise an error message appears when starting.

Creating a symbol for projects

Follow these steps to create a project symbol:

Step	Action
1	Under **Start** → **Settings** → **Taskbar...**, you can open the **Taskbar Properties** dialog box.
2	In the register **Start Menu Programs**/**Expanded** (Win2000), select the **Add...** command button.
3	In the **Create Shortcut** dialog box, select the **Browse...** command button.
4	In the **Browse** dialog box, go to the Concept installation path and double-click on the file **CONCEPT.EXE**. **Result:** The **Browse** dialog box is closed and the file **CONCEPT.EXE** is entered, including the path, in the **Command line:** text box, e.g. **C:\CONCEPT\CONCEPT.EXE**.
5	Now add the project path and project name to the command line, e.g. **C:\CONCEPT\CONCEPT.EXE PLANT1.PRJ** and confirm the entry using **Next>** command button. **Note:** To create a connection to any PLC, add additional Parameters (See *Automatic Connection with Command Line Parameters (Modbus, Modbus +, TCP/IP), p. 1080*) to the command line.
6	In the **Select program group** dialog box, select an existing program group for the symbol or create a new one using **New folder...**. Confirm the entry using the **Next>** command button.
7	In the **Select program designation** dialog box, select the project name and confirm using the **Finish** command button.
8	Close the **Taskbar Properties** dialog box with **OK**. **Result:** The properties dialog box is closed and the project symbol is available in the start menu of the folder you selected.
9	Open the folder with the project symbol in the Start Menu. Select the project symbol and click the right mouse button. **Result:** A menu window is opened.

Step	Action
10	Select the **Properties** command button. **Result:** The **"Project Symbol Name" Properties** dialog box is opened.
11	Go to the **Connection** register and complete the command line **Working directory**/**Target** (Win2000) with the name of the project directory, e.g. **C:\CONCEPT\PROJECTS**. Confirm the entry using the **Set** command button.
12	Then exit the dialog box by selecting **OK**.
13	Open the project by clicking on the project symbol.

Creating a symbol for DFBs

In this way, symbols can also be created for DFBs. To do this, select the file **CCEPTDFB.EXE** in step 4 and add the DFB name and path instead of the project name and path in step 5.

B.5 Online help

At a Glance

Overview

This section describes use of online help.

What's in this Section?

This section contains the following topics:

At a Glance

General information

The online help is used to quickly and easily obtain information about the task being performed, the use of an unfamiliar command or the functions, Function Blocks and modules.

The online help is available throughout Concept.

> **Note:** The option **Use polygon acceleration** may not be used if the graphics card has hardware acceleration functions. Use of these may still lead to the graphics in the online help being incomplete. A detailed description of how to switch off the acceleration function will be found in the graphics card's user manual.

Starting the online help

There are several methods of calling up the online help:
- Invoking the contents)
  There are two methods of invoking the online help contents:
  - To invoke the online help contents, select the menu command **Help** → **Contents**.
  - In the program group Concept open the help symbol.
- Help with the execution of a menu command
  There are two methods of invoking help with a menu command:
  - using the mouse)
    To obtain an explanation select the menu command with the left mouse button, hold down the mouse button, press **F1**, and then release the mouse button.
  - using the keyboard)
    To obtain an explanation of a menu command, select it and then press **F1**.
- Help with a dialog
  There are two methods of invoking help with a dialog:
  - To obtain an explanation of a dialog, click on the command button **Help** in the dialog itself.
  - To obtain an explanation of a dialog, press **F1**in the dialog itself.
- Help with operating an EFB
  To obtain an explanation of the operation of the EFB, click on the command button **Help with type** within the dialog with the EFB properties.
- Help with the operation of a module
  In the dialog **I/O module selection** click on the command button **Help with module**, to obtain an explanation of the operation of a module.

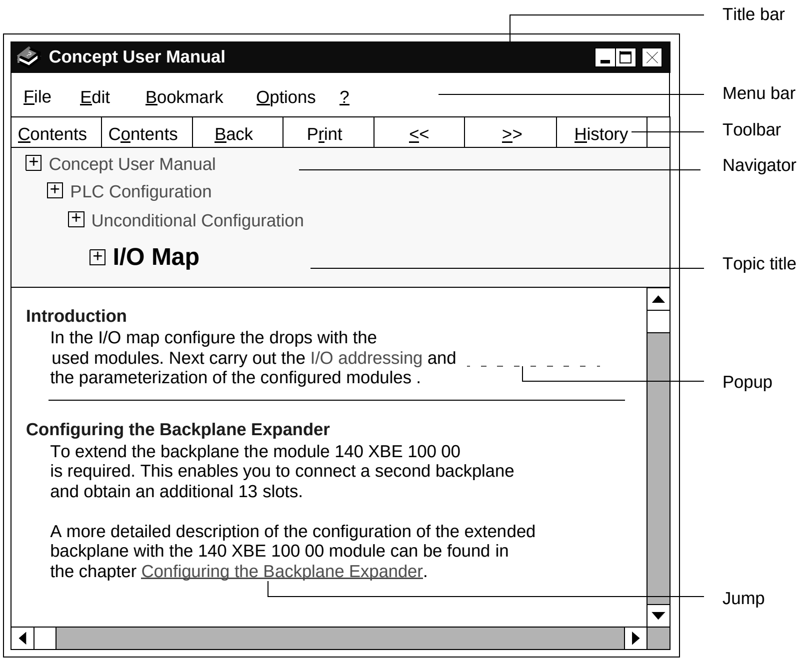
How the Online Help is set out

Introduction

If you start the online help, the Windows Help system opens, containing either
- a table of contents (if you started with **Help** → **Contents** or the icon),
- or containing a description of the dialog (if you started with the **Help** command button),
- or containing a description of an EFB (if you started with the **Help on Type** command button),
- or containing a description of a module (if you started with the **Module Help** command button),

This section describes the Concept specific basics of the online help window.

Online help window:



Title Bar

The title bar contains the active help file names, or in other words the help project.

Menu Bar

A description of the standard menu bar can be found in the respective Microsoft Windows manual.

Toolbar

The following buttons are available in Concept:

- Contents
  This button is used to invoke the online help contents directory.
  Details about this function can be found in the corresponding Windows Manual.
  Note: If you jump (See *Jump, p. 758*) between different help projects and click the **Contents** button, the contents of the invoked help project (rather than the current one) is displayed. This is a Microsoft error. The Navigator is available to allow you to navigate within the current help project (related topics *Navigator, p. 758*).
- Index
  This button is used to invoke an index for finding help texts.
  Details about this function can be found in the corresponding Windows Manual.
  Note: If you want to carry out a search of the whole text, press the **Index** command button, select the **Search** index card, choose the desired search function and type in the term you’re looking for.
- Back
  This button is used to invoke the previously read help text.
- Print
  This button is used to print out the current topic (the current help topic).
- <<
  This button is used to "browse" the previous help text. This button is used to read the online help like a book. When you have reached the first "page" of the online help (contents directory), the button is hidden.
- >>
  This button is used to "browse" to the next help text. This button is used to read the online help like a book. When you have reached the last "page" of the online help, the button is hidden.
- **History**
  When you use this button a window opens which displays all of the help topics that are already open.

Title of Topic

The topic title refers to the title of a chapter from paper documentation. This topic title always remains visible, even if, in the case of long documents, the text is moved in the window.

Navigator

The Navigator is in the topic title. It serves as a navigator inside the help projects.

Jump

A jump can be recognized by the fact it is written in green and is underlined. When you click on a jump, the help text corresponding to this key word/ topic appears. Jumps correspond to "related topics" entries in paper documents, the pages are however removed for your convenience. The invoked help text is then replaced by a new help text.

Popup

A popup can be recognized by the fact it is written in green and has a dotted line under it. When you click on a popup, the help text corresponding to this key word appears. Popups correspond to glossary entries in paper documents, however, the pages here are removed for your convenience. To display the text, a popup window is opened. This popup window may contain further popups. The popup window is cleared by re-clicking on it or pressing any key. This does not replace the present help text.

List of symbols and short cut keys

C

At a Glance

Description

Each editor and the PLC configuration have their own list of symbols available. This facilitates access to frequently used functions. It is also possible to call up many functions with short cut keys instead of menu commands.

What's in this Chapter?

This chapter contains the following sections:

C.1 Icon bar

At a Glance

Description

This section describes the icon bar icons. In the icon bars there are editor independent and editor dependent icons.

What's in this Section?

This section contains the following topics:

General icon bar

Symbols

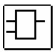
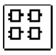
The table below shows the available symbols and their corresponding menu entry commands:

Symbol	Menu entry command executed
	**File → Open...**
	**File → New section... / New DFB section...**
	**File → Open section...**
	**File → Save**
	**Project → Variable declaration...**
	**Project → Search…**
	**Online → Online control panel...**
	**Online → Download changes...**
	**Edit → Undo: Delete**
	**Edit → Cut**
	**Edit → Copy**
	**Edit → Paste**

Icon bar in the FBD editor

Symbols

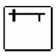
The table shows the additional icons available in the FBD editor and the corresponding menu entry commands (see also *General icon bar, p. 764*):

Symbol	Menu entry command executed
	**View → Zoom in**
	**View → Zoom out**
	**Objects → Select**
	**Objects → Link**
	**Objects → FFB: Last Type**
	**Objects → Invert Input/Output**
	**Objects → Text**
	**Objects → FFB selection...**
	**Online → Animate selected**
	**Online → Animate booleans**

Icon bar in the SFC-Editor

Symbols

The table shows the additional icons available in the SFC editor and the corresponding menu entry commands (see also *General icon bar, p. 764*):

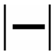
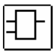
Symbol	Menu entry command executed
	**View → Zoom in**
	**View → Zoom out**
	**Objects → Select**
	**Objects → Select column structure**
	**Objects → Select row structure**
	**Objects → Step**
	**Objects → Transition**
	**Objects → Parallel branch**
	**Objects → Parallel joint**
	**Objects → Alternative branch**
	**Objects → Alternative joint**
	**Objects → Jump**
	**Objects → Link**
	**Objects → Step - Transition sequence**
	**Objects → Structured Parallel sequence**
	**Objects → Structured Alternative sequence**

Symbol	Menu entry command executed
	**Objects → Transition - Step sequence**
	**Objects → Text**
	**Online → Animate**
	**Online → Animation Panel functions**

Icon bar in the LD editor

Symbols

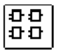
The table shows the additional symbols available in the LD editor and the corresponding menu entry commands (please also refer to the *General icon bar, p. 764*):

Symbol	Menu entry command executed
	**View → Zoom in**
	**View → Zoom out**
	**Objects → Select**
	**Objects → Link**
	**Objects → Direct Link**
	**Objects → Vertical Link**
	**Objects → FFB: Last Type**
	**Objects → Invert Input/Output**
	**Objects → Text**
	**Objects → FFB selection...**
	**Objects → Coil**
	**Objects → Coil - Negated**
	**Objects → Contact - Normally Open**
	**Objects → Contact – Normally Closed**
	**Online → Animate selected**
	**Online → Animate booleans**

List of Symbols in the IL and ST Editor

Symbols

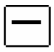
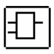
The table shows the additional symbols available in the IL and ST editor and the corresponding menu entry commands (see also *General icon bar, p. 764*):

Symbol	Menu Entry Command Executed
	**Objects → Insert FFB**
	**Online → Watch Selected**
	**Online → Animate booleans**

List of Symbols in the LL984-Editor

Symbols

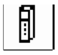
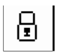
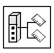
The table shows the additional symbols available in the LL984 editor and the corresponding menu entry commands (see also *General icon bar, p. 764*):

Symbol	Menu Entry Command Executed
	**Objects → Select**
	**Objects → Coil**
	**Objects → Coil - Retentive**
	**Objects → Horiz Short**
	**Objects → Vertical Short**
	**Objects → Contact – Normally Open**
	**Objects → Contact – Normally Closed**
	**Objects → Contact – Pos Trans**
	**Objects → Contact – Neg Trans**
	**Objects → Instruction: Last Type**
	**Objects → List Instructions...**

Icons in PLC Configuration

Icons

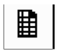
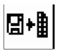
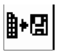
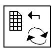
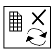
The table shows the icons also available in PLC configuration and their allocated menu commands (related topics: *General icon bar, p. 764*):

Icon	Executed menu command
	**PLC configuration → PLC Selection...**
	**PLC configuration → Memory Partitions...**
	**PLC configuration → ASCII Setup...**
	**PLC configuration → Loadables...**
	**PLC configuration → Config. Extension...**
	**PLC configuration → Segment scheduler...**
	**PLC configuration → I/O Map...**
	**PLC configuration → Data Protection...**
	**PLC configuration → Peer Cop...**
	**PLC configuration → Ethernet / I/O Scanner...**
	**PLC configuration → Hot Standby...**
	**PLC configuration → ASCII Port Settings...**
	PLC configuration → Modbus Port Settings...
	**PLC configuration → Specials...**

Toolbar in the RDE Editor

Icons

The table shows the icons also available in the RDE Editor and their allocated menu commands (see also *General icon bar, p. 764*):

Icon	Executed menu command
	**Template → New Template...**
	**Template → Open Template...**
	**Template → Save Template**
	**Online → Animate**
	**Online → Download Reference Data**
	**Online → Get CSL**
	**Online → Delete CSL**

Toolbar in the Project Browser

Icons

The table shows the additional symbols available in the project browser and the corresponding menu commands (also see *General icon bar, p. 764*):

Icon	Menu command executed
	**Project Shortcut Menu → Animate Enable States**
	**Project Shortcut Menu → Show Detailed View**

C.2 Short cut keys

At a Glance

Description

This section describes the available short cut keys. There are editor independent and editor dependent short cut keys.

What's in this Section?

This section contains the following topics:

General Short Cut Keys

Short Cut Keys

The table shows the short cut keys available and the corresponding menu entry command:

Short Cut Keys	Menu Entry Command Executed
F1	Calls the context-sensitive online help. Use this key to call up an explanation of the menu entry command or dialog chosen. In dialogs, this key corresponds to the menu entry command **Help**.
Ctrl+F4	**System menu (for the document window) → Close document window**
Ctrl+F6	**System menu (for the document window) → Next**
Ctrl+S	**File → Save project/save DFB**
Alt+F4	**File → Quit the application window (Concept-Application)**
F8	**Project → Variable declarations...**
F3	**Project → Search**
Shift+F3	**Project → Trace**
F5	**Project → Search history...**
F6	**Project → Search next**
Alt+F9	**Project → Analyze section**
Ctrl+P	**Online → Online control panel...**
F9	**Online → Single sweep trigger**
Ctrl+R	**Online → Reference Data Editor**
Shift+F5	**Window → Cascade**
Shift+F4	**Window → Tile Vertically**

Short Cut Keys in the IL, ST and Data Type Editor

Calling up menu command entries

The table shows the short cut keys available in the IL, ST and Data Type Editor and the corresponding menu entry commands (see also *General Short Cut Keys, p. 774*):

Key	Menu Entry Command Executed
Ctrl+Z	**Edit → Undo delete**
Ctrl+X	**Edit → Cut**
Ctrl+C	**Edit → Copy**
Ctrl+V	**Edit → Paste**
Del	**Edit → Delete**
Ctrl+G	**Edit → Goto line...**
Ctrl+J	**Edit → Goto counterpart**
Ctrl+E	**Edit → Expand statement**
Alt+F8	**Edit → Lookup variables**
Ctrl+F	**Edit → Find next**
Ctrl+H	**Edit → Replace...**
Ctrl+Y	**Online → Animate Booleans**
Ctrl+I	**Online → Inspect Selected**
Ctrl+W	**Online → Watch Selected**

Moving insertion marks in the text

Moving insertion marks in the text:

Key	Moving
Down	Onto the next line
Up	Onto the previous line
Ctrl+G	Onto a specific line
End	To the end of the line
Home	To the beginning of the line
Picture up	Into the next window
Picture up	Into the previous window
Ctrl+Right	To the next word
Ctrl+Left	To the previous word
Ctrl+End	To the end of the document
Ctrl+Home	To the beginning of the document

Deleting text

Deleting text:

Key	Function
Backspace Key (Delete backwards)	Deleting a mark (or deleting marked text) to the left of the insertion mark.
Del	Deleting a character (or deleting marked text) to the right of the insertion mark.
Ctrl+**Backspace key** (Delete backwards)	Deleting a line

Marking text

Marking text:

Key	Extending the marking
Shift+**Right**	to the next character
Shift+**Left**	to the previous character
Ctrl+**Shift**+**Right**	to the next word
Ctrl+**Shift**+**Left**	to the previous word
Shift+**Down**	to the next line
Shift+**Up**	to the previous line
Shift+**End**	to the end of the line
Shift+**Home**	to the beginning of the line
Shift+**Picture down**	to a window underneath
Shift+**Picture up**	to a window above
Ctrl+**Shift**+**Picture down**	to the end of the current window
Ctrl+**Shift**+**Picture up**	to the beginning of the current window
Ctrl+**Shift**+**End**	to the end of the document
Ctrl+**Shift**+**Home**	to the beginning of the document

Editing text

Editing text:

Key	Function
Ctrl+**X**	Deleting marked text and saving in the clipboard
Ctrl+**C**	Copying marked text and saving in the clipboard
Entering the new text	Replacing marked text
Del	Deleting marked text without saving in the clipboard
Ctrl+**V**	Replacing marked text with text from the clipboard.
Ctrl+**F**	Searching for text
Ctrl+**R**	Replacing text

Short Cut Keys in the FBD and SFC Editor

At a Glance

Concept supports the work with the keyboard in the graphic editors. Although the mouse is a more appropriate input tool, it is nevertheless possible to operate Concept with the keyboard alone – especially in machine environments. The editors behave in the same way regardless of whether they are operated with the mouse or with the keyboard.

Rules

The following general rules need to be observed:

- The space bar corresponds to the left mouse button, i.e. the space bar is used for selecting and moving.
- The enter key corresponds to the double click with the left mouse button – for example, the input key is used to call up the properties dialog of objects.
- The shift key is used in conjunction with the keyboard exactly as it is with the mouse – for example, the shift key is used to extend an object selection or to reselect a few objects from a number which have already been selected.

Calling up menu command entries

The table shows the short cut keys available in the FBD and SFC editor and the corresponding menu entry commands (see also *General Short Cut Keys, p. 774*):

Key	Menu Entry Command Executed
Ctrl+**A**	**Edit** → **Select All**
Ctrl+**Z**	**Edit** → **Undo delete**
Ctrl+**X**	**Edit** → **Cut**
Ctrl+**C**	**Edit** → **Copy**
Ctrl+**V**	**Edit** → **Paste**
Del	**Edit** → **Delete**
Ctrl+**O**	**View** → **Overview**
Ctrl+**N**	**View** → **Normal**
Ctrl+**E**	**View** → **Expanded** (only in SFC)
Ctrl+**+**	**View** → **Zoom in**
Ctrl+**-**	**View** → **Zoom out**
Ctrl+**Y**	In the FBD Editor: **Online** → **Animate booleans** In SFC-Editor: **Online** → **Animate**
Ctrl+**W**	**Online** → **Animate selected** (in FBD)

Moving the cursor

Moving the cursor:

Key	Function
Cursor keys	The **cursor keys** move the cursor inside the document window. The cursor is moved further around a Pixel. If the cursor is at the edge of the document window, pressing the cursor keys again will page the document window in the corresponding direction.
Ctrl+**Cursor Keys**	When the **Strg** key is pressed, the **cursor keys** move the cursor inside the document window. The cursor is moved further around a logical unit (depending on the active editor). If the cursor is at the edge of the document window, pressing the **cursor keys** again will page the document window in the corresponding direction
Home	The **Pos1** key moves the cursor to the left-hand edge of the document window.
End	The **End** key moves the cursor to the right-hand edge of the document window.

Scrolling

Scrolling:

Key	Function
Ctrl+**Home**	When the **Ctrl** key is pressed, the **Pos1** key moves the document window to the upper left-hand corner of the section.
Ctrl+**End**	When the **Ctrl** key is pressed, the **End** key moves the document window to the lower right-hand corner of the section.
Picture up	The **picture up** key scrolls the document window one screen page upwards, while the cursor remains in the same position in the document window.
Picture down	The **picture down** key scrolls the document window one screen page downwards, while the cursor remains in the same position in the document window.
Ctrl+**Picture up**	When the **Ctrl** key is pressed, the **Picture up** key scrolls the document window one page to the left while the cursor remains in the same place in the document window.
Ctrl+**Picture down**	When the **Ctrl** key is pressed, the **Picture down** key scrolls the document window one page to the right while the cursor remains in the same place in the document window.

Edit

Edit

Key	Function
Space bar	In select mode, the object at the cursor position is selected and all other objects are deselected. In placing mode the corresponding object is placed where the cursor is.
Shift key+**Space bar**	In selection mode, when the **Shift** key is pressed, objects which have not previously been selected in the cursor position are selected, or vice versa. The selection of all other objects is not affected. In placing mode the corresponding object is placed where the cursor is.
Space bar+**Cursor Keys**	In selection mode – if there is no selected object where the cursor is – the cursor moves and a selection rectangle is displayed. If a selected object is in the cursor position, all objects will be shifted according to how the cursor is moved. The number of inputs of an FFB with a variable input number can be changed in the FB Editor's Selection Mode by placing the cursor on the rectangle in the middle of the lower edge of the selection frame, which holds down the **Space bar** and presses the **Up** or **Down** keys. The width of the branches or connections can be changed in the SFC Editor's Selection Mode by placing the cursor on the rectangle of the selection frame, which holds down the **Space bar** and presses the **Right** or **Left** keys. In Link Mode, a link is produced by dragging the mouse.
Shift key+**Space bar**+**Cursor keys**	In Selection Mode, this key combination creates a selection frame as described above, and the selection of all other objects is retained.

Allocating variables onto an FFB

To allocate variables onto an FFB, do the following:

Step	Action
1	Use the **cursor keys** or **Shift**+**cursor keys** to move the cursor to the input/ output of the FFB.
2	Press **Enter**. **Reaction:** The **Connect FFB** dialog for the selected input/output opens.

Changing variables onto an FFB

To change variables onto an FFB, do the following:

Step	Action
1	Use the **cursor keys** or **Shift**+**cursor keys** to move the cursor to the FFB variables to be changed.
2	Press **Enter**. **Reaction:** The **Connect FFB** dialog for the selected input/output opens.

Changing the number of inputs/outputs

To change the number of inputs/outputs with extendable FFBs, do the following:

Step	Action
1	Use the **cursor keys** or **Shift**+**cursor keys** to move the cursor to the centre of the lower edge of the FFB's block frame.
2	Press **Space bar**+**Down cursor key** to generate further inputs/outputs. Press **Space bar**+**Up cursor key** to hide further inputs/outputs. **Reaction:** The number of inputs/outputs is changed.

Shortcut keys in the LD-Editor

At a Glance

Concept supports the work with the keyboard in the graphic editors. Although the mouse is a more appropriate input tool, it is nevertheless possible to operate Concept with the keyboard alone – especially in machine environments. The Editors behave in the same way regardless of whether they are operated with the mouse or with the keyboard.

Rules

The following general rules need to be observed:
- The space bar corresponds to the left mouse button, i.e. the space bar is used for selecting and moving.
- The Enter key corresponds to the double click with the left mouse button – for example, the input key is used to call up the properties dialog of objects.
- The Shift key is used in conjunction with the keyboard exactly as it is with the mouse – for example, the Shift key is used to extend an object selection or to reselect a few objects from a number which have already been selected.
- Pressing a key only once only affects the element in the center of the current cell.
- Pressing a key together with **Ctrl** affects the right side of the current cell..
- Striking a key together with **Shift** afects the left side of the current cell

Calling up menu command

The table shows the additional shortcut keys and their corresponding menu commands avialable in LD Editor (see also *General Short Cut Keys, p. 774*):

Key	Menu Entry Command Executed
Ctrl+A	**Edit → Select All**
Ctrl+Z	**Edit → Undo delete**
Ctrl+X	**Edit → Cut**
Ctrl+C	**Edit → Copy**
Ctrl+V	**Edit → Paste**
Del	**Edit → Delete**
Ctrl+O	**View → Overview**
Ctrl+N	**View → Normal**
Ctrl++	**View → Zoom in**
Ctrl+-	**View → Zoom out**
Esc	**Objects → Select**
Shift+H	**Objekts → Link**
H	**Objects → Direct Link**
V	**Objects → Vertical Link**
F	**Objects → FFB: Last Type**
I	**Objects → Invert Input/Output**
T	**Objects → Text**
Shift+F	**Objects → FFB selection...**
C	**Objects → Contact Normally Open**
L	**Objects → Contact – Normally Closed**
P	**Objects → Contact - Rising Edge (Positive)**
N	**Objects → Contact - Falling Edge (Negative)**
Shift+C	**Objects → Coil**
Shift+L	**Objects → Coil - Negated**
Shift+S	**Objects → Coil - Set**
Shift+R	**Objects → Coil - Reset**
Shift+P	**Objects → Coil - Rising Edge (Positive)**
Shift+N	**Objects → Coil - Falling Edge (Negative)**
Ctrl+Y	**Online → Animate booleans**
Ctrl+W	**Online → Animate selected**

Placing objects

In order to place objects in the LD Editor by using the keyboard, please carry out the following steps:

Step	Action
1	Move the field with a gray background onto the field where the object is to be placed (move gray field (selecting a field)).
2	Strike the key assigned to the object (see *Creating objects, p. 786*). **Reaction:** Adjoining boolean objects are automatically connected.
3	Links between non-adjoining objects and non-boolean in/outputs have to be made with the mouse pointer (see *Moving the mouse pointer, p. 785*).
4	The mouse pointer must also be used to invert in/outputs (see *Moving the mouse pointer, p. 785*).

Moving the gray field (selecting a field)

Moving the gray field (selecting a field)

Key	Function
Up	Moves the gray field up by one field
Down	Moves the gray field down by one field
To the right	Movesthe gray fields to the right by one field
To the left	Moves the gray fields to the left by one field
Home	Moves the gray field to the left margin
Shift+**Home**	Moves the gray field to the left margin
End	Moves the gray field to the right margin
Shift+**End**	Moves the gray field to the right margin
Ctrl+**Home**	Moves the gray field to the top left-hand corner
Ctrl+**End**	Moves the gray field to the top right-hand corner

Selecting objects

Selecting objects

Key	Function
Space character	Selects object in the middle of the gray field
Ctrl+**Space character**	Selects object on the right-hand side of the gray field
Shift+**Space character**	Selects object on the left-hand side of the gray field
Enter	In select mode: Selects object in the middle of the gray field and opens its Select dialog (if available)
Ctrl+**Enter**	In select mode: Selects object from the right-hand side of the gray field and opens its Select dialog (if available)
Shift+**Enter**	In select mode: Selects object from the left-hand side of the gray field and opens its Select dialog (if available)

Moving a selected object

Moving a selected object

Key	Function
Shift+**Up**	Moves the selected object up by one field
Shift+**Down**	Moves the selected object down by one field
Shift+**Right**	Moves the selected object to the right by one field
Shift+**Left**	Moves the selected object to the left by one field

Allocating variables onto an FFB

To allocate variables onto an FFB, do the following:

Step	Action
1	Move the gray field onto the cell containing the in/output.
2	To allocate variables to inputs, press **Ctrl**+**Enter**. To allocate variables to outputs press **Ctrl**+**Enter**. **Reaction:** The dialog **Connect FFB** of the selected in/output is opened.

Changing variables onto an FFB

To change variables onto an FFB, do the following:

Step	Action
1	Move the gray field onto the cell containing the variable to be changed.
2	To select the variable press **Shift**+**Enter**. **Reaction:** The dialog **Connect FFB** of the selected in/output is opened.

Deleting vertical links

To delete vertical variables, carry out the following step:

Step	Action
1	Move the gray field onto the cell running through the vertical link.
2	Press **Ctrl**+**Delete**. **Reaction:** The vertical link is deleted.

Moving the mouse pointer

Moving the mouse pointer

Key	Function
Ctrl+**Up**	Moving the mouse pointer up by one step
Ctrl+**Down**	Moving the mouse pointer down by one step
Ctrl+**Right**	Moving the mouse pointer to the right by one step
Ctrl+**Left**	Moving the mouse pointer to the left by one step

Scrolling

Scrolling:

Key	Function
Picture up	Scrolls the display sector one page up
Shift+**Picture up**	Scrolls the display sector one page up
Picture down	Scrolls the display sector one page down
Shift+**Picture down**	Scrolls the display sector one page down
Ctrl+**Picture up**	Scrolls the display sector one page to the right
Ctrl+**Picture down**	Scrolls the display sector one page to the right

Creating objects

Creating objects

Key	Function
C	Creates a N.O. in the gray field
L	Creates an opener in the gray field
P	Creates a contract for the recognition of positive flanks in the gray field
N	Creates a contract for the recognition of negative flanks in the gray field
Shift+**C**	Creates a coil in the gray field
Shift+**L**	Creates a negated coil in the gray field
Shift+**S**	Creates a coil set in the gray field
Shift+**R**	Creates a reset coil in the gray field
Shift+**P**	Creates a coil for the recognition of positive flanks in the gray field
Shift+**N**	Creates a coil for the recognition of negative flanks in the gray field
Shift+**F**	Opens FFB selection dialog
F	Creates current FFB in the gray field

Creating links

Creating links

Key	Function
H	Activates the link mode
V	Creates a vertical link in the right-hand bottom corner of the gray field (and then moves the gray field to the right by one field)
Shift+**V**	Creates a vertical link in the bottom left-hand corner of the gray field.

Activating the different modes

Activating the different modes

Key	Function
Space character	Activates the selection mode
Esc	Activates the selection mode
H	Activates the link mode
I	Activates the mode for inverting in/outputs
T	Activates the text mode

Short Cut Keys in the LL984-Editor

Short Cut Keys

The table shows the additional short cut keys available in the LL984 editor and the corresponding menu entry commands (see also *General Short Cut Keys, p. 774*):

Short Cut Keys	Menu Entry Command Executed
Ctrl+Z	Edit → Undo delete
Ctrl+X	Edit → Cut
Ctrl+C	Edit → Copy
Ctrl+V	Edit → Paste
Del	Edit → Delete
Ctrl+D	Edit → DX Zoom...
Ctrl+H	Edit → Offset References...
Ctrl+O	View → Overview
Ctrl+N	View → Normal
Ctrl+E	View → Expanded
Ctrl++	View → Zoom in
Ctrl+-	View → Zoom out
(	Objects → Coil
Ctrl+L	Objects → Coil - Retentive
"	Objects → Contact – Normally Open
/	Objects → Contact – Normally Closed
P	Objects → Contact – Pos Trans
N	Objects → Contact – Neg Trans
=	Objects → Horiz Short
I	Objects → Vertical Short
Ctrl+F	Objects → Instruction by name...
Ctrl+G	Network → Goto...
Ctrl+I	Networks → Insert
Ctrl+Q	Networks → Insert Equation
Ctrl+A	Networks → Append
Ctrl+U	Networks → Attach formula
Ctrl+K	Networks → Delete
Picture up	Networks → Next
Picture up	Networks → Previous
Ctrl+M	Networks → Comment...

Short Cut Keys	Menu Entry Command Executed
Ctrl+T	**Online → Trace**
Ctrl+B	**Online → ReTrace**

IEC conformity

D

At a Glance

Overview

This Chapter contains the standards tables required by IEC 1131-1.

What's in this Chapter?

This chapter contains the following sections:

D.1 What is the IEC 1131-3 standard?

At a Glance

Overview

This section contains general information about IEC 1131-3 and the implemented IEC conformity test.

What's in this Section?

This section contains the following topics:

General information about IEC conformity

At a Glance

The IEC standard 1131-3 (compare chapter 1.4) specifies the syntax and semantics of a standardized series of programming languages for Programmable Logic Controls (PLC). These include the two text languages IL (Instruction List) and ST (Structured Text) and the two graphical languages LD (Ladder Diagram) and FBD (Function Block Diagram).

It also defines the elements of the sequential function chart (SFC) language for structuring the internal organization of PLC programs and Function Blocks. Configuration elements, used for installing PLC programs onto PLC systems, are also defined.

Note: Concept uses the English acronyms for the programming languages.

Furthermore, it defines methods to enable communication between the PLC and other automated system components.

Concept standard accordance

In accordance with the standard, the present version of the programming system Concept supports a subset of language elements, which are defined in the standard.

In this context, accordance with the standard means the following:
- The standard allows the individual implementing an IEC program system to select or deselect certain language properties or even complete languages from the selection tables, which represent an integrated part of the standard specifications. A system, which itself accords with the standard, may only implement the selected properties exactly as they are given in the standard.
- In addition, the standard enables the individual implementing to introduce defined language elements into an interactive programming environment. As the standard expressly emphasizes that the specification of such environments lies outside of its area of application, the person implementing has a certain degree of freedom to offer optimized forms of display and implementation mechanisms for the benefit of the user.
- Concept uses these degrees of freedom e.g. when introducing the term "Project" to implement the IEC language elements "Configuration", "Resource" and "Program" all together (Concept only supports one single cyclically running program within a single resource within the configuration). Apart from this, it uses them, for example, with implementation mechanisms made available for declaring variables and authorizing Function Blocks.

IEC standards tables

Information on which properties are supported and other implementation specific details can be found in the following statements on standard fulfilment and the associated standards tables.

IEC Conformity Test

Testing the Import/Export Interface

An interface for importing standard IEC programs and DFBs from ASCII files (menu **File** → **Import**) and exporting these programs into graphical languages in ASCII format (menu **File** → **Export**) is available in Concept. The conformity of this interface can be tested using files which can be obtained from IFAK (Institut für Automation und Kommunikation e.V. Magdeburg).

IEC conformity test scripts:
(c) 1994, IFAK Institut für Automation und Kommunikation e.V.
Magdeburg
Steinfeldstraße 3
D-39179 Barleben

Notes

The following points must be considered with regard to the conformity of the import interface:

- In Concept, IL operators are permitted as identifiers.
  R, S, LD, S1 and R1 are possible parameter names. Therefore, there will be no changes made to the standard functions/function blocks. Concept requires no change in the IEC table 54 with S to SET, R to RESET, S1 to SET1, R1 to RESET1.
- All IL operators not in conflict with functions are permitted as variable names in Concept (N, S, R, S1, R1, CLK, CU, CD, PV, IN, PT) – contrary to IEC table 54.
- Counter EFBs must be typified in Concept, e.g. CTU must become CTU_INT.
- Function block instances cannot be called up more than once; a restriction that is self-evident if IEC table 53, property 3 is required.
- An overflow of time span variables (e.g. t#100s) is not detected. The system calculates the time correctly, so that detection of an overflow is not necessary.
- IEC IL comments are only permitted as the last element in a line. Concept allows comments to be made everywhere.

D.2 IEC standards tables

At a Glance

Overview

This system fulfils the requirements of the IEC 1131-3 in the following properties of the language.

What's in this Section?

This section contains the following topics:

Common elements

IEC standards table

IEC standards table for common elements:

Table number	Property number	Property description
1	1	For required character set – see *Chapter 2.1.1* of 1131-3
1	2	Lower case characters
1	3a	Hash key (#)
1	4a	Dollar sign ($)
1	5a	Vertical line (|)
1	6a	Left and right square brackets "[ ]"
2	1	Upper case character and numbers
2	2	Upper and lower case characters, numbers, embedded underscore
2	3	Upper and lower case characters, numbers, leading and embedded underscore
3	1	Comments
4	1	Integer (whole number) literals
4	2	Real literals
4	3	Real literals with exponents
4	4	Base 2 literals
4	5	Base 8 literals
4	6	Base 16 literals
4	7	Boolean zero and one
4	8	Boolean FALSE and TRUE
7	1a	Time span without underscores: short prefix
7	1b	Time span without underscores: long prefix
7	2a	Time span with underscores short prefix
7	2b	Time span with underscores long prefix
10	1	BOOL: Boolean
10	3	INT: Integer
10	4	DINT: Double integer
10	7	UINT: Signed integer
10	8	UDINT: Signed double integer
10	10	REAL: Floating point number
10	12	TIME: Time span

Table number	**Property number**	**Property description**
10	17	BYTE: Bit sequence 8
10	18	WORD: Bit sequence 16
12	4	Data types for fields
12	5	Data types for structures
15	1	I: Input (*Note 1, p. 800*)
15	2	Q: Output (*Note 2, p. 800*)
15	4	X: Bit size (*Note 2, p. 800*, *Note 1, p. 800*)
15	5	no prefix: Bit size (*Note 2, p. 800*, *Note 1, p. 800*)
15	6	B: Byte size (*Note 2, p. 800*, *Note 1, p. 800*)
15	7	W: Word size (*Note 2, p. 800*, *Note 1, p. 800*)
15	8	D: Double word size (*Note 2, p. 800*, *Note 1, p. 800*)
17	2	Declaration of directly displayed buffered variables (*Note 5, p. 800*, *Note 9, p. 801*)
17	3	Declaration of storage locations with symbolic variables (*Note 5, p. 800*)
17	4	Assignment of storage locations with fields (*Note 5, p. 800*, *Note 11, p. 801*)
17	5	Automatic storage allocation for symbolic variables (*Note 5, p. 800*)
17	7	Declaration for buffered fields (*Note 5, p. 800*, *Note 11, p. 801*)
17	8	Declaration for structured variables (*Note 5, p. 800*)
18	2	Initialization of directly displayed buffered variables (*Note 5, p. 800*, *Note 9, p. 801*, *Note 10, p. 801*)
18	3	Assignment of storage locations and start values for fields(*Note 5, p. 800*)
18	4	Assignment of storage locations and start values for fields (*Note 5, p. 800*, *Note 11, p. 801*)
18	5	Initialization of symbolic variables (*Note 5, p. 800*)
18	7	Declaration and initialization of buffered variables (*Note 5, p. 800*, *Note 11, p. 801*)
18	8	Initialization of structured variables (*Note 5, p. 800*)
18	9	Initialization of constants
19	1	Negated input
19	2	Negated output
20	1	Use of "EN" and "ENO" - REQUIRED for LD (*Note 6, p. 800*)
20	2	Use of "EN" and "ENO" – OPTIONAL for FBD

Table number	Property number	Property description
20	3	FBD without "EN" and "ENO"
21	2	Standardized functions (*Note 3, p. 800*)
22	1	(*-TO-**) Type conversion functions (*Note 4, p. 800*
22	2	Truncation towards zero: TRUNC (*Note 3, p. 800*))
23	1	ABS: Absolute value
23	2	SQRT: Square root
23	3	LN: Natural logarithm
23	4	LOG: Base 10 logarithm
23	5	EXP: Exponential function
23	6	SIN: Sine, input in radians
23	7	COS: Cosine, input in radians
23	8	TAN: Tangent, input in radians
23	9	ASIN: Arc sine, principal value
23	10	ACOS: Arc cosine, principal value
23	11	ATAN: Arc tangent, principal value
24	12	ADD: Add
24	13	MUL: Multiply
24	14	SUB: Subtract
24	15	DIV: Divide
24	16	MOD: Modulo
24	17	EXPT: Exponentiation
24	18	MOVE: Assignment
25	1	SHL: move to the left
25	2	SHR: Move to the right
25	3	ROR: Rotate to the right
25	4	ROL: Rotate to the left
26	5	AND: LLogical And
26	6	OR: Logical Or
26	7	XOR Logical exclusive Or
26	8	NOT: Negation
27	1	SEL: Binary selection
27	2a	MAX: Extendable maximum
27	2b	MIN: Extendable minimum
27	3	LIMIT: Limit

Table number	**Property number**	**Property description**
27	4	MUX: Extendable multiplexer
28	5	GT: Falling sequence
28	6	GE: Monotonic sequence (decreasing)
28	7	EQ: Equality
28	8	LE: Monotonic sequence (increasing)
28	9	LT: Rising seqence
28	10	NE: Inequality
30	1	ADD: Adding TIME to TIME
30	4	SUB: Subtracting TIME from TIME
30	10	MUL: Multiplying TIME by ANY_NUM
30	11	DIV: Dividing TIME by ANY_NUM
33	1	RETAIN identifier for internal variables (*Note 5, p. 800*)
33	2	RETAIN identifier for output variables (*Note 5, p. 800*)
33	3	RETAIN identifier for internal Function Blocks (*Note 5, p. 800*)
34	1	Bistable Function Block (set priority)
34	2	Bistable Function Block (reset priority)
35	1	Detecting the rising edge
35	2	Detecting the falling edge
36	1	Up counter
36	2	Down counter
36	3	Up/Down counter
37	1	TP: Pulse (timer)
37	2a	TON: Switch-on delay
37	3a	TOF: Switch-off delay
39	1	RETAIN identifier for internal variables (*Note 5, p. 800*)
39	2	RETAIN identifier for output variables (*Note 5, p. 800*)
39	3	RETAIN identifier for internal Function Blocks (*Note 5, p. 800*)
39	14	Assignment of storage locations with fields (*Note 5, p. 800*)
39	18	Assignment of storage locations and start values for fields(*Note 5, p. 800*)
39	19	Use of directly displayed variables (*Note 2, p. 800*, *Note 1, p. 800*)
40	1	Step/Start step – graphical form with directional links
40	2	Step/Start step – text form without directional links (Note 8)
40	3a	Step marker – general form

Table number	**Property number**	**Property description**
40	4	Step time elapsed – general form
41	1	Transition condition in ST language within the graphic (*Note 8, p. 801*)
41	5	Transition condition in ST language – textual reference (*Note 9, p. 801*)
41	6	Transition condition in IL language – textual reference (*Note 9, p. 801*)
41	7	Use of the transition name
41	7b	Transition condition in FBD language
41	7c	Transition condition in IL language
41	7d	Transition condition in ST language
42	1	Each Boolean variable can be an action
43	1	Action block
43	2	Concatenated action blocks
43	3	Step body in text form (*Note 8, p. 801*)
44	1	Identifier
44	2	Action name
45	1	Not saved (no identifier)
45	2	N: not saved
45	3	R: Overriding reset
45	4	S: Set (saved)
45	5	L: Time limited
45	6	D: Delayed
45	7	P: Pulse
45	9	DS: Delayed and saved
46	1	Simple string
46	2a	Branching in string selection (priority from left to right)
46	3	Merging a string selection
46	4	Parallel strings - branch and merge
46	5a	String jump (priority from left to right)
46	6a	String loop (priority from left to right)

Note 1

Modicon TSX Quantum Präfix 3 is used in the prefix IB, ID position in all graphical languages.

Note 2

Modicon TSX Quantum Präfix 4 is used in the prefix QB, QD position in all graphical languages.

Note 3

The following functions are overloaded with reference to the data which is selected, multiplexed or assigned; the type statement refers to the selection parameters.

List of overloaded functions:
- SEL
- MUX
- MOVE

All other functions are standardized, e.g. REAL_TRUNC_INT.

Note 4

List of type conversion functions:
- BOOL_TO_BYTE, BOOL_TO_DINT, BOOL_TO_INT, BOOL_TO_REAL, BOOL_TO_TIME, BOOL_TO_UDINT, BOOL_TO_UINT, BOOL_TO_WORD,
- BYTE_TO_BOOL, BYTE_TO_DINT, BYTE_TO_INT, BYTE_TO_REAL, BYTE_TO_TIME, BYTE_TO_UDINT, BYTE_TO_UINT, BYTE_TO_WORD,
- DINT_TO_BOOL, DINT_TO_BYTE, DINT_TO_INT, DINT_TO_REAL, DINT_TO_TIME, DINT_TO_UDINT, DINT_TO_UINT, DINT_TO_WORD,
- INT_TO_BOOL, INT_TO_BYTE, INT_TO_DINT, INT_TO_REAL, INT_TO_TIME, INT_TO_UDINT, INT_TO_UINT, INT_TO_WORD,
- REAL_TO_BOOL, REAL_TO_BYTE, REAL_TO_DINT, REAL_TO_INT, REAL_TO_TIME, REAL_TO_UDINT, REAL_TO_UINT, REAL_TO_WORD,
- TIME_TO_BOOL, TIME_TO_BYTE, TIME_TO_DINT, TIME_TO_INT, TIME_TO_REAL, TIME_TO_UDINT, TIME_TO_UINT, TIME_TO_WORD,
- UDINT_TO_BOOL, UDINT_TO_BYTE, UDINT_TO_DINT, UDINT_TO_INT, UDINT_TO_REAL, UDINT_TO_TIME, UDINT_TO_UINT, UDINT_TO_WORD,
- UINT_TO_BOOL, UINT_TO_BYTE, UINT_TO_DINT, UINT_TO_INT, UINT_TO_REAL, UINT_TO_TIME, UINT_TO_UDINT, UINT_TO_WORD,
- WORD_TO_BOOL, WORD_TO_BYTE, WORD_TO_DINT, WORD_TO_INT, WORD_TO_REAL, WORD_TO_TIME, WORD_TO_UDINT, WORD_TO_UINT

The consequences of each conversion are described in the block library and the help texts, which are available for the library of IEC standard functions.

Note 5

The RETAIN identifier is implicitly required; no language elements displayed in non-buffered memory areas are supported.

Note 6

"EN" and "ENO" are offered as standard; they can, however, be hidden and any other input or output of data type BOOL can be used for links.

Note 7	Expressions are restricted to individual simple Boolean variables.
Note 8	Only available on import of IEC text form in graphical SFC representation.
Note 9	Only available in textual declaration in IL or ST sections.
Note 10	Initialization only possible for non Boolean outputs.
Note 11	Declaration of field variables only possible when using previously defined field data type names.

IL (AWL) language elements

IEC standards table

IEC standards table for IL (AWL) language elements:

Table number	Property number	Property description
52	1	LD operator: sets the current result to that of the operand
52	2	ST operator: saves the current result to the operand address
52	3	S operator: sets Boolean operands to "1" R operator: sets Boolean operands to "0"
52	4	AND operator
52	6	OR operator
52	7	XOR operator
52	8	ADD operator
52	9	SUB operator
52	10	MUL operator
52	11	DIV operator
52	12	GT operator: Comparison >
52	13	GE operator: Comparison >=
52	14	EQ operator: Comparison =
52	15	NE operator: Comparison <>
52	16	LE operator: Comparison <=
52	17	LT operator: Comparison <
52	18	JMP operator: Jump to tag (*Note 1, p. 802*)
52	19	CAL operator: Calls Function Block
52	21	Closing bracket ")": Editing deferred operations
53	1	CAL operator with list of input parameters
53	2	CAL operator with loading/saving of input parameters

Note 1

Jumps are only allowed within sections, not across section boundaries.

Note 2

The following keywords are not available:
- TYPE...END_TYP
- VAR_INPUT...END_VAR
- VAR_OUTPUT...END_VAR
- VAR_IN_OUT...END_VAR
- VAR_EXTERNAL...END_VAR
- FUNCTION...END_FUNCTION
- FUNCTION_BLOCK...END_FUNCTION_BLOCK
- PROGRAM...END_PROGRAM
- STEP...END_STEP
- TRANSITION...END_TRANSITION
- ACTION...END_ACTION
- SEGMENT_SCHEDULER
- RET
- &

ST language elements

IEC standards table

IEC standards table for ST language elements:

Table number	Property number	Property description
55	1	Placing in brackets: (Expression)
55	2	Function calls: Function name (list of arguments)
55	3	Exponentiation: **
55	4	Negation: -
55	5	Complement: NOT
55	6	Multiplication: *
55	7	Division: /
55	8	Modulo: MOD
55	9	Addition: +
55	10	Subtraction: -
55	11	Comparison: <, >, <=, >=
55	12	Equality: =
55	13	Inequality: <>
55	14	Boolean AND: &
55	15	Boolean AND: AND
55	16	Boolean exclusive OR: XOR
55	17	Boolean OR: OR
56	1	Assignment
56	2	Function Block calls and use of FB outputs
56	4	IF instruction
56	5	CASE instruction
56	6	FOR instruction
56	7	WHILE instruction
56	8	REPEAT instruction
56	9	EXIT instruction
56	10	Empty instruction

Note 1

The following keywords are not available:
- TYPE...END_TYP
- VAR_INPUT...END_VAR
- VAR_OUTPUT...END_VAR
- VAR_IN_OUT...END_VAR
- VAR_EXTERNAL...END_VAR
- FUNCTION...END_FUNCTION
- FUNCTION_BLOCK...END_FUNCTION_BLOCK
- PROGRAM...END_PROGRAM
- STEP...END_STEP
- TRANSITION...END_TRANSITION
- ACTION...END_ACTION
- SEGMENT_SCHEDULER
- RETURN

Common graphic elements

IEC standards table

IEC standards table for common graphic elements:

Table number	Property number	Property description
57	2	Horizontal lines: Graphic or semi-graphic
57	4	Vertical lines: Graphic or semi-graphic
57	6	Horizontal/vertical connection: Graphic or semi-graphic
57	8	Line intersection without connection: Graphic or semi-graphic
57	10	Connected and unconnected corners: Graphic or semi-graphic
57	12	Blocks with connecting lines: Graphic or semi-graphic

LD (KOP) language elements

IEC standards table

IEC standards table for LD (KOP) language elements:

Table number	Property number	Property description
59	1	Left power rail (with linked horizontal connection)
60	1	Horizontal connection
60	2	Vertical connection (with linked horizontal connections)
61	1	Closer
61	3	Opener
61	5	Contact for detection of positive transition
61	7	Contact for detection of negative transition
62	1	Coil (*Note 1, p. 807*)
62	2	Negative coil (*Note 1, p. 807*)-{}-
62	3	SET coil (*Note 1, p. 807*)
62	4	RESET coil (*Note 1, p. 807*)
62	8	Coil for detection of positive transition
62	9	Coil for detection of negative transition

Note 1

In start behavior of PLCs there is a distinction between cold starts and warm starts:

- **Cold start**
  Following a cold start (loading the program with **Online** → **Load**) all variables (irrespective of type) are set to "0" or, if available, their initial value.
- **Warm start**
  In a warm start (stopping and starting the program or **Online** → **Load changes**) different start behaviors are valid for located variables/direct addresses and unlocated variables:
  - **Located variables/direct addresses**
    In a warm start all 0x, 1x and 3x registers are set to "0" or, if available, their initial value.
    4x registers retain their current value (storage behavior).
  - **Unlocated variables**
    In a warm start all unlocated variables retain their current value (storing behavior).

This varying behavior in a warm start leads to peculiarities in the warm start behavior of set and reset functions.

- **Set and Reset in LD and IL**
  Warm start behavior is dependent on the variable type used (storage behavior in use of unlocated variables; non storage behavior in use of located variables/ direct addresses)
- **SR and RS Function Blocks in FBD, LD, IL and ST**
  These Function Blocks work with internal unlocated variables and therefore always have a storage behavior.

Implementation-dependent parameters

IEC standards table

IEC standards table for implementation-dependent parameters:

Parameters	Threshold values/behavior
Error-handling procedure	See *Error causes, p. 811* & EFB help
National characters used	All characters in the Windows ANSI character set are supported.
Maximum length of identifiers	Program name: 8 Formal parameter names: 8 DFB type names: 8 EFB type names: 17 Data type names: 24 all others: 32
Maximum comment length:	Limited only by Windows resources
Range of values for time span literals	0s to 49d_17h_2m_47.295s
Range of values for variables of type TIME	0s to 49d_17h_2m_47.295s
Accuracy of the seconds display with types TIME_OF_DAY and DATE_AND_TIME	not applicable
Maximum number of field indices	Practically no limit
Maximum field size	64 kB
Maximum number of structure elements	Only limited by Windows or PLC resources
Maximum structure size	64 kB
Maximum number of variables per declaration	Only limited by Windows or PLC resources
Maximum number of enumerated values	not applicable
Default maximum length of STRING variables	not applicable
Maximum authorized length of STRING variables	not applicable
Maximum number of hierarchy tiers	1
Configured or physical illustration	Configured illustration, physical illustration through separate I/O projection
Parameters	Threshold values/behavior
Maximum number of indices	Practically no limit
Maximum range of index values	Range of data type INT

Parameters	Threshold values/behavior
Maximum number of structure levels	Only limited by Windows or PLC resources
Initialization of system inputs	System zero; no user-definable start values
Maximum number of variables per declaration	Only limited by Windows or PLC resources
Information for the determination of execution times of program organization units	In preparation
Methods of function display (names or symbols)	Names
Maximum number of function specifications	not applicable
Maximum number of inputs for extendable functions	32
Type conversion accuracy	See EFB help
Accuracy of functions of a variable	INTEL floating point processor or emulator
Arithmetic function implementation	INTEL floating point processor or emulator
Maximum number of Function Block specifications	Only limited by Windows or PLC resources
Maximum number of Function Block authorizations	512 per section; number of sections per program organization unit is only limited by Windows or PLC resources
Pvmin, Pvmax of counters	Limited by rangess of the INT or DINT data types
Effect of a change in the value of a PT input during a time measurement operation	Directly affects the timer's default time
Program size limits	Only limited by available PLC memory
Time behavior and porting effects of the execution control elements	The execution of SFC networks in different sections occurs sequentially, in the order given in these sections.
Accuracy of elapsed step time	10 ms
Maximum number of steps per SFC	Limited by the available area for entering characters within the section; number of sections per program organization unit only limited by Windows or PLC resources; the upper limit for the total number of objects per SFC is 2000

Parameters	Threshold values/behavior
Parameters	Threshold values/behavior
Maximum number of transitions per SFC and per step	Limited by the available area for entering characters within the section; number of sections per program organization unit only limited by Windows or PLC resources; the upper limit for the total number of objects per SFC is 2000
Action control mechanism	Functionally equivalent to the specification in the standard
Maximum number of actions per step	Only limited by Windows or PLC resources
Graphical display of the step situation	Green = active Red = inactive
Transition switch time	Of the magnitude of 10 ms
Maximum width of branches/connections	Limited by the available area for entering characters 32
Contents of the RESOURCE libraries	See EFB libraries & help
Maximum number of tasks	1
Task interval resolution	not applicable
Pre-justified and non pre-justified schedules	not applicable
Maximum length of expressions	Practically no limit
Partial evaluation of Boolean expressions	no partial evaluation
Maximum length of instructions	Practically no limit
Maximum number of CASE selections	Practically no limit
Value of the control variables on completion of FOR loops	undefined
Graphic/semi-graphic display	Graphic
Network topology restrictions	no restrictions
Evaluation sequence of feedback loops	Within a network, the starting point of the FFB execution sequence is determined by the "single" available feedback variable

Parameters	Threshold values/behavior
Means of specifying the network execution sequence	1: Execution sequence of program organization unit sections 2: The network execution sequence can be changed within sections; this is done by using a menu command to switch between the execution sequences of two selected FFB items

Error causes

IEC standards table

IEC standards table for error causes:

Error cause	Handling (see *Note 1, p. 812*)
Variable value exceeds the specified range	not applicable
Initialization list length and number of field elements do not agree	2) Error message during programming
Incorrect use of directly displayed or external variables in functions	not applicable
Type conversion error	4) Error message during execution
Numerical result exceeds the range for data type	4) Error message during execution
Division by zero	4) Error message during execution
Mixed input data types in a selection function	2) Error message during programming
Selector (K) outside MUX function range	4) Error message during execution
Invalid character position	not applicable
Result exceeds maximum sequence length	not applicable
Numerical result exceeds the range for data type	4) Error message during execution
Zero or more than one starting step in SFC network	3) Error message during analysis/loading/ connection
User program attempting to change step situation or step time	2) Error message during programming
Simultaneously completed transitions without priority in a selection branch	not applicable
Side effects of evaluation of a transition condition	3) Error message during analysis/loading/ connection

Error cause	**Handling (see *Note 1, p. 812*)**
Action control error	1) Error not reported
Unsafe or unreachable SFCs	3) Error message during analysis/loading/ connection
Data type conflict in VAR_ACCESS	not applicable
Tasks demanding too many processor resources	3) Error message during analysis/loading/ connect
Scan time overrun	4) Error message during execution
Error cause	Handling (see note 1)
Further task schedule conflicts	not applicable
Numerical result exceeds the range for data type	4) Error message during execution
Division by zero	4) Error message during execution
Invalid data type for operation	3) Error message during analysis/loading/ binding
Return from function without assigned value	not applicable
Occurrence arrives at no outcome	4) Error message during execution
The same identifier as connector tag and element name use	not applicable
Non-initialized feedback variable (initialized with system zero)	1) Error not reported

Note 1

Identification for the handling of error causes according to IEC 1131-3, chapter 1.5.1, d):
- 1) Error not reported
- 2) Error message during programming
- 3) Error message during analysis/loading/binding
- 4) Error message during execution

D.3 Expansions of IEC 1131-3

Expansions of IEC 1131-3

At a Glance

The Concept programming environment makes the construct of the so-called section available in all programming languages permitting the subdivision of a program organization unit. This construct provides the opportunity to mix several languages in the body of a POU (e.g. FBD sections, SFC sections), a property, which, if used for this purpose, represents an expansion of the IEC syntax. Sections do not generate their own name space; the name space for all language elements is the POU.

Sections appearing in the body of a POU written only in the FBD language are not to be viewed as an expansion, rather as a permitted means of specifying the execution sequences of several FBD networks furnished with tags, as specified in the corrigendum to 1131-3.

Purpose of sections

Sections serve various purposes

- Sections permit the functional division of an expansive POU body: The body of a POU can be divided into sensible functional parts. The section list represents a kind of functional table of contents for a large, otherwise unstructured POU body.
- Sections permit the graphical division of an expansive POU body: in accordance with an intentionally graphic form of representation, sub-structures of an expansive body can be established. Smaller or larger partial structures may be chosen.
- The division of an expansive POU body enables faster online changes: the section serves as the unit for online changes in Concept. If the POU body is changed in various places during the program runtime, all sections affected by the changes are taken into account if explicitly initiated reloading occurs.
- Sections permit the execution sequence to influence particular marked parts of the POU body: the section name serves as a marking for the part of the body contained in the section, and the execution sequence of the sections can be changed by ranking the sections (see also the last part of the "implementation-dependent parameters" table for information on the execution sequence of networks in the FBD language).
- Sections permit the parallel use of different languages in the same POU: this property is a considerable expansion of the syntax of the IEC 1131-3 standard, which only permits the use of a single IEC language for a POU body. Only the SFC language also provides the opportunity to formulate parts of the body in different languages, because transitions and actions can be expressed in any language, in as far as the corresponding properties are supported by the programming system.

D.4 Text language syntax

Text Language Syntax

Description

The programming system Concept supports the complete language syntax, as specified in appendix B of the IEC language standard 1131-3, with the following exceptions:

- Syntax productions in appendix B of 1131-3, belonging to properties, which according to the IEC standards tables in *IEC standards tables, p. 794* in this document are not supported by Concept, are not implemented.
- The use of some Concept supported properties is, according to the associated remarks in the IEC standards table, only possible in a restricted or modified form. The associated syntax productions are therefore only occasionally or somewhat differently implemented.
- Concept supports the NOT Operator for inverting Boolean battery content in IL.
- The implementation of some faulty syntax productions in appendix B of 1131-3, improved upon either in the corrigendum to 1131-3 or in the planned amendment to 1131-3, uses the suggestions in these documents for orientation.
  The improved productions are implemented in Concept as follows (chapter numbers refer to appendix B of 1131-3):
  - **B.1.3.3:**
    ```
    array_initialization ::= '[' array_initial_elements {','
    array_initial_elements} ']'

    initialized_structure ::= structure_type_name [':='
    structure_initialization]
    ```
  - **B.2.1:**
    ```
    il_operand_list ::= il_operand [',' [EOL] il_operand]

    il_fb_call ::= ('CAL' | 'CALC' | 'CALCN') fb_name '('
    il_operand_list ')'
    ```
  - **B.2.2:**
    ```
    il_operator ::= 'LD' | 'LDN' | 'ST' | 'STN' | 'S' | 'R'
    | ('AND' | 'ANDN' | 'OR' | 'ORN' | 'XOR' | 'XORN') ['(']
    | ('ADD' | 'SUB' | 'MUL' | 'DIV') ['(']
    | ('GT' | 'GE' | 'EQ' | 'NE' | 'LT' | 'LE') ['(']
    | 'JMP' | 'JMPC' | 'JMPCN' | ')' | function_name
    ```

Index

A

D

E

H

I

J

K

L

P

Q

R

S

T

U

V

W