DriveWare®

User Manual
DriveSPC

DriveSPC

User Manual

Drive*Ware*®

Code: 3AFE 68836590 REV K EN

EFFECTIVE: 1.6.2009

FIDRI\ENT000
PDM code: 00564796.DOC

Table of contents

Introduction to the manual

Chapter overview

This chapter includes a description of the contents of the manual. It also contains information about the intended audience and related publications.

Reader

The reader of this manual is expected to know the block programming principle, the use of tasks (time levels) in real-time programming, data types, Boolean operations and arithmetic functions.

Contents

The chapter *Introduction to DriveSPC* briefly lists the main features of the DriveSPC program.

The chapter *Installation of DriveSPC* gives information on installing the DriveSPC program.

The chapter *Solution programming* describes the Solution Programming method, including function blocks, custom circuits, pins, time levels, user parameters, alarms, faults, and template files.

The chapter *Use of DriveSPC* describes the usage of the DriveSPC program, including user interface, program modification actions, and the On-Line mode.

The chapter *Menu commands (main program window)* describes the menu commands of the main program window.

The chapter *Menu commands (CC window, function blocks)* describes the menu commands of the custom circuit (function blocks) definition window.

The chapter *Menu commands (CC window, Structured Text)* describes the menu commands of the custom circuit (Structured Text) definition window.

The *Appendix 1* summarises the use of the Ctrl, Alt, Shift, and F3 keys.

The *Appendix 2* describes the use of arrays and structures in solution programs.

The *Appendix 3* describes the Parameter Manager window.

The *Appendix 4* describes the Alarm Manager window.

The *Appendix 5* describes the Fault Manager window.

The *Appendix 6* describes the Structured Text language implementation of DriveSPC.

The *Appendix 7* describes the additional features of the Pro version.

Related documents

DriveStudio User Manual

ACSM1 Base Program Firmware Manual

Introduction to DriveSPC

General

DriveSPC (**S**olution **P**rogram **C**omposer) is a Windows-based tool for Solution Programming. It can be used without a drive or with one or more drives.

There are two Solution Programming (SP) methods available:

- *Function blocks*
- *Structured Text (ST) language*

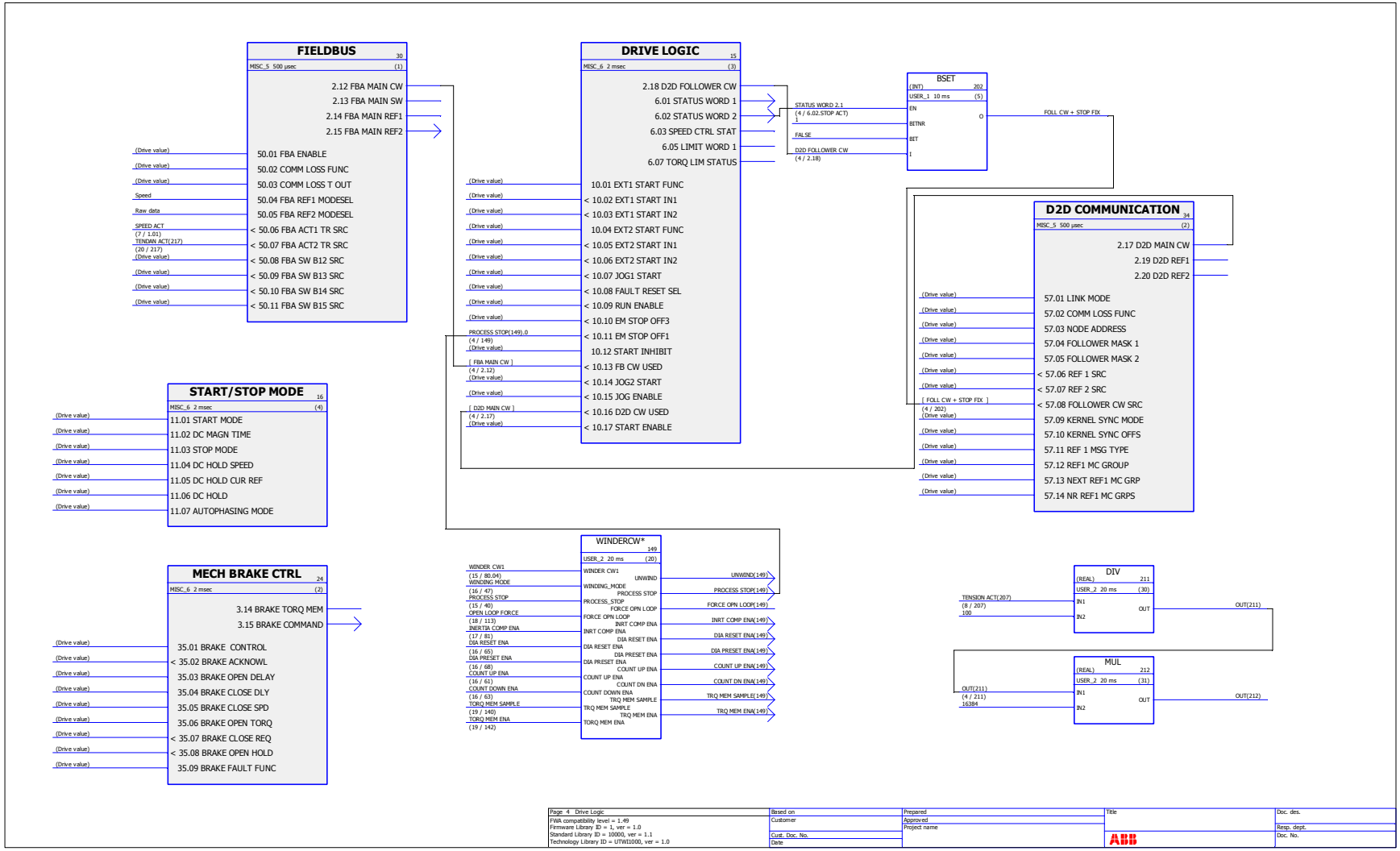
These programming methods are described in the IEC 61131-3 standard.

Solution programs execute in drives.

With DriveSPC, it is possible to:

- make a connection to a drive. If more than one drive is connected to the PC, a separate DriveSPC window can be opened for each of them.
- upload the solution program from the connected drive and display it graphically on the screen. All program information exists in the drive; no additional PC files are required.
- monitor actual values of the user-selected block pins and Structured Text variables on the screen in real-time
- modify the solution program on the screen
- utilise hierarchical programming
- define new parameters, parameter groups, alarms, and faults
- make several program protections with passwords
- download the program to the connected drive and start the execution of the program in the drive
- print program pages
- save the program to a disk file
- open a saved program file and display it graphically on the screen. The program can be modified, downloaded, saved, and so on.
- compare programs
- see descriptions of all blocks and block-related parameters

A page of a solution program is shown on the next page.


FIELDBUS
30
MISC_5 500 µsec (1)
2.12 FBA MAIN CW
2.13 FBA MAIN SW
2.14 FBA MAIN REF1
2.15 FBA MAIN REF2
50.01 FBA ENABLE
50.02 COMM LOSS FUNC
50.03 COMM LOSS T OUT
50.04 FBA REF1 MODESEL
50.05 FBA REF2 MODESEL
50.06 FBA ACT1 TR SRC
50.07 FBA ACT2 TR SRC
50.08 FBA SW B12 SRC
50.09 FBA SW B13 SRC
50.10 FBA SW B14 SRC
50.11 FBA SW B15 SRC
DRIVE LOGIC
15
MISC_6 2 msec (3)
2.18 D2D FOLLOWER CW
6.01 STATUS WORD 1
6.02 STATUS WORD 2
6.03 SPEED CTRL STAT
6.05 LIMIT WORD 1
6.07 TORQ LIM STATUS
10.01 EXT1 START FUNC
10.02 EXT1 START IN1
10.03 EXT1 START IN2
10.04 EXT2 START FUNC
10.05 EXT2 START IN1
10.06 EXT2 START IN2
10.07 JOG1 START
10.08 FAULT RESET SEL
10.09 RUN ENABLE
10.10 EM STOP OFF3
10.11 EM STOP OFF1
10.12 START INHIBIT
10.13 FB CW USED
10.14 JOG2 START
10.15 JOG ENABLE
10.16 D2D CW USED
10.17 START ENABLE
BSET
(INT) 202
USER_1 10 ms (5)
EN
BITNR
BIT
O
I
STATUS WORD 2.1
(4 / 6.02.STOP ACT)
1
FALSE
D2D FOLLOWER CW
(4 / 2.18)
FOLL CW + STOP FIX
D2D COMMUNICATION
34
MISC_5 500 µsec (2)
2.17 D2D MAIN CW
2.19 D2D REF1
2.20 D2D REF2
57.01 LINK MODE
57.02 COMM LOSS FUNC
57.03 NODE ADDRESS
57.04 FOLLOWER MASK 1
57.05 FOLLOWER MASK 2
57.06 REF 1 SRC
57.07 REF 2 SRC
57.08 FOLLOWER CW SRC
57.09 KERNEL SYNC MODE
57.10 KERNEL SYNC OFFS
57.11 REF 1 MSG TYPE
57.12 REF1 MC GROUP
57.13 NEXT REF1 MC GRP
57.14 NR REF1 MC GRPS
START/STOP MODE
16
MISC_6 2 msec (4)
11.01 START MODE
11.02 DC MAGN TIME
11.03 STOP MODE
11.04 DC HOLD SPEED
11.05 DC HOLD CUR REF
11.06 DC HOLD
11.07 AUTOPHASING MODE
MECH BRAKE CTRL
24
MISC_6 2 msec (2)
3.14 BRAKE TORQ MEM
3.15 BRAKE COMMAND
35.01 BRAKE CONTROL
35.02 BRAKE ACKNOWL
35.03 BRAKE OPEN DELAY
35.04 BRAKE CLOSE DLY
35.05 BRAKE CLOSE SPD
35.06 BRAKE OPEN TORQ
35.07 BRAKE CLOSE REQ
35.08 BRAKE OPEN HOLD
35.09 BRAKE FAULT FUNC
WINDERCW*
149
USER_2 20 ms (20)
WINDER CW1
WINDING_MODE
PROCESS_STOP
FORCE OPN LOOP
INRT COMP ENA
DIA RESET ENA
DIA PRESET ENA
COUNT UP ENA
COUNT DN ENA
COUNT DOWN ENA
TRQ MEM SAMPLE
TRQ MEM ENA
UNWIND
PROCESS STOP
DIV
(REAL) 211
USER_2 20 ms (30)
IN1
IN2
OUT
MUL
(REAL) 212
USER_2 20 ms (31)
IN1
IN2
OUT
Page 4 Drive Logic
FWA compatibility level = 1.49
Firmware Library ID = 1, ver = 1.0
Standard Library ID = 10000, ver = 1.1
Technology Library ID = UTWI1000, ver = 1.0
Based on
Customer
Cust. Doc. No.
Date
Prepared
Approved
Project name
Title
Doc. des.
Resp. dept.
Doc. No.
ABB

Operating modes

Off-Line mode

- This is the default operating mode after the startup of DriveSPC
- The mode can be used without or with a drive
- In the *Off-Line* mode, it is always possible to:
  - open a program file and display it graphically on the screen
  - modify the program
  - print program pages
  - save the program to a disk file
- If one or more drives exist, it is also possible to:
  - connect to the user-selected drive
  - upload the program from the connected drive and display it
  - download the program to the connected drive and start the execution of the program in the drive
  - remove the program from the connected drive
  - go to the *On-Line* mode

On-Line mode

- All value modifications of block-related parameters are automatically copied to the connected drive
- Actual values of user-selected block pins and Structured Text variables are displayed on the screen in real-time

Installation of DriveSPC

DriveStudio must be installed before installing DriveSPC.

DriveSPC is designed to run under the Microsoft Windows XP, 2000, or Vista operating environment on IBM-compatible PCs.

Note: You must have Administrator privileges to your PC.

Insert the DriveSPC CD-ROM into the CD drive of your PC.

Start the *Control Panel* program (**Start > Settings > Control Panel**), and double click the *Add/Remove Programs* icon.

Click the **Add New Programs** button, and follow the instructions that appear on the screen (select file *Setup.exe* from the CD-ROM):

Solution Programming

Chapter overview

This chapter describes the Solution Programming method.

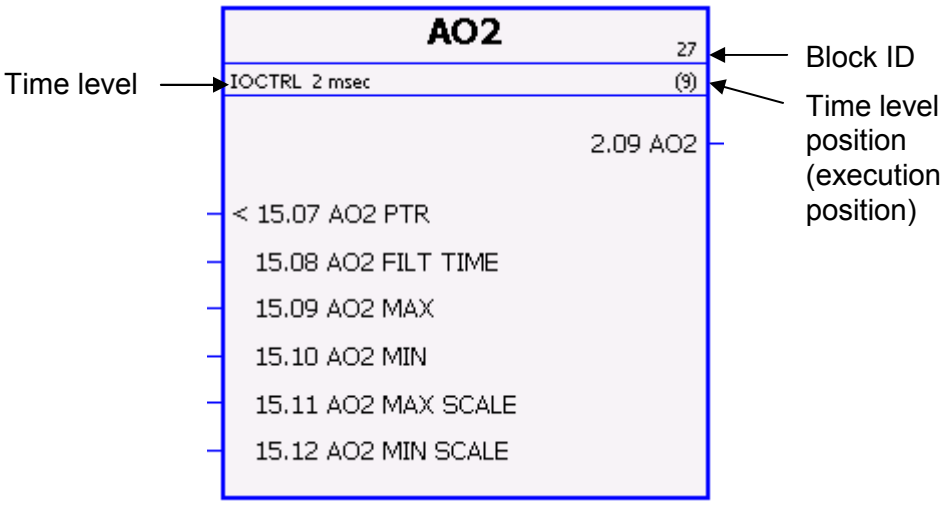
Function blocks

Solution Programming is based on the use of function blocks.
Structured Text language can be used for the creation of new blocks.

Blocks can be divided into two main groups:

- *Firmware blocks* are parts of the drive control firmware connectable by pin parameters.
  One instance of every firmware block always exists in every solution program. It is not possible to add or remove firmware block instances.

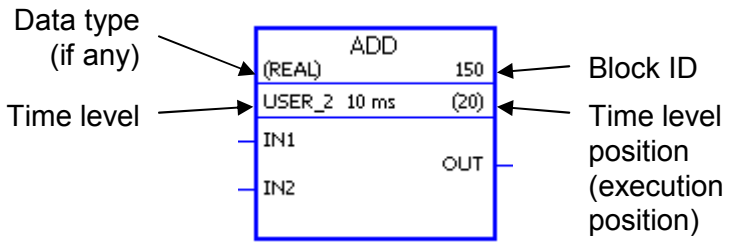
  Firmware block example:



  Block IDs are assigned automatically by DriveSPC and their values never change. All block IDs are unique.
  Time levels are explained below. Time level position (9) above means that this block is the 9th block of time level IOCTRL.

- *Normal blocks* can be divided into two groups:
  - *Standard blocks* always exist, and one or more instances of these blocks can be added to all solution programs.
  - *Technology blocks* are optional blocks. Their instances can be used in a solution program only if the appropriate technology library is in the drive. New technology blocks can be made on demand.

Normal block example:



Block IDs are assigned automatically by DriveSPC, and their values never change. All block IDs are unique.
Time levels are explained below. Time level position (20) above means that this block is the 20th block of time level USER_2.

Data types of some normal blocks (such as the block above) are user-selectable when instances of these blocks are added to solution programs.

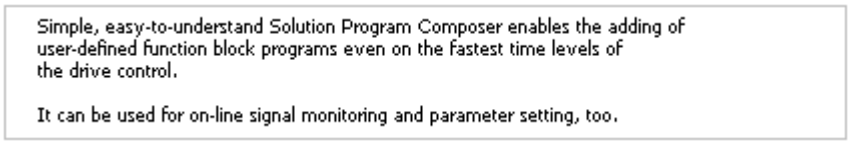
Some normal blocks have extensible pins. The user can specify how many copies of extensible pins are added to the block when an instance of the block is added to a program.

If a block is inside a conditional IF-ELSE branch, a small magenta square ▫ is displayed in its upper left corner.

In addition to the function blocks mentioned above, there are three virtual block types ("virtual" means that these blocks are not actually executed when a solution program is running in a drive):

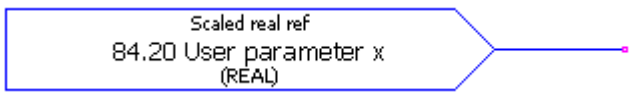
- *Comment* block for user's comment texts.

  Comment block example:



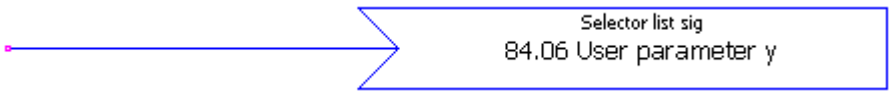
- *Reference parameter* block for user parameters that can be written by users.

  Reference parameter block example:



- *Signal parameter* block for user parameters that can only be read by users.

  Signal parameter block example:



Custom circuits

In *Hierarchical programming*, several blocks with their connections can be represented as a single block.

Hierarchical programming in DriveSPC is based on the use of user-defined custom circuits.

A **custom circuit** (**CC**) contains either a user-defined collection of one or more function block instances and their user-defined pin connections or a user-written Structured Text program.

After a CC has been defined, one or more instances of it can be added to one or more solution programs just as with instances of normal function blocks.

Definition of custom circuits

Definition with function blocks:

The programming of a CC definition is made in the *CC definition window* in the same way as the actual solution programming is made in the solution program window.

Definition with the Structured Text language:

The programming of a CC definition is made in the *CC definition window* by writing a textual program in the Structured Text language (see *Appendix 6*).

External pins:

Every CC definition must include one or more user-specified **external pins**. These are input and/or output pins or variables in the CC definition that will be visible in the instances of the CC definition. External pins make it possible to connect a CC instance to other blocks of the solution program.

Instances of the defined custom circuits

Instances of the created CC definitions can be added to the solution program that was in the solution window when these CC definitions were made.

If you want to add instances of a created CC definition to other solution programs, save the CC definition to a disk file.
Now you can open another solution program, read the CC file into this program, and add instances of the read CC definition to this program.

Instances of CC definitions look similar to normal function blocks and they are added to solution programs in the same way as instances of normal blocks are added to programs.

Changing used custom circuit definitions

If you have used instances of a CC definition in your solution program, you can still make changes to this definition later on, as long as you do not change the definitions of external pins.

If you want to change the *data type* of an external pin in a CC definition and this pin is connected in one or more instances of this CC definition, you have to disconnect this pin in these instances before the pin change.

If you want to change the *number* or *input/output type* of external pins, you have to remove all instances of this CC definition from the solution program and add the instances again after you have changed the external pins in the CC definition.

Custom circuit archive folder

It is suggested that all custom circuit definition files are saved to the same folder.

The pathname of the folder that is used by the CC file save or read operation is saved to the solution program. This folder is initially displayed by the next CC file save/read operation.

Time levels

Solution programs are *real-time program*s. Real-time programming is based on *task*s, and the drive software contains several tasks. Tasks are called *time levels* here, and they have names like `TRQREF`, `IOCTRL`, and `USER_1`.
The length, or the execution interval, of some time levels is fixed, but the length of the other time levels can be changed by the user.

Every block and CC instance of a solution program resides in some time level. The execution order and the time level of every *firmware* block instance is always fixed, but the execution order and the time level of a *normal* block instance and a *CC* instance is specified by the user.

There is a predefined *priority order* between time levels. If the execution of the blocks of a time level is not completed when a higher priority time level is scheduled to start its execution, the execution of these lower priority blocks is suspended until the blocks of the higher priority time level have been executed.

***Warning:* It is very important that the total execution time of all blocks of a time level does not exceed the length, or the execution interval, of this time level.**
If it does, the drive will fault after the download of the program.

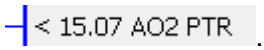
Pins

Input and output pins (connection points) of blocks are marked with a small magenta bubble.

Pin parameters of firmware blocks

The input and output pins of all firmware blocks are parameters.

The type of an input pin parameter can be one of the following:

- *Value*.
  This parameter is usually set to a constant value, but some value parameters can be connected to an output pin, too.
- *Value pointer*
  Marked with the '<' character inside the block, for example, 

  .
  This parameter is usually connected to an output pin, but some value pointer parameters can be set to a constant value, too.
- *Bit pointer*
  Marked as value pointer above.
  This parameter is either connected to a bit of an output pin or it is set to the constant Boolean value `FALSE` or `TRUE`.

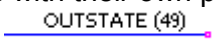
The user-selectable value type of an input pin parameter can be one of the following:

- *Drive value* (text `Drive value` is displayed in the pin)
  The value of this pin parameter is not specified in the solution program.
  When the program is downloaded, the current drive value of this parameter is preserved during the download operation.
- *Locked value* (e.g. `1.25`)
  When the program is downloaded, this value or pin connection is set to the drive parameter.
  Locked parameter values can not be changed in the drive after download.
- *Unlocked value* (value is enclosed in brackets, e.g. `[ 1.25 ]`)
  When the program is downloaded, the user can select one of the following two alternatives:
  - All unlocked values and pin connections are set to drive parameters.
  - The current drive values of all these parameters are preserved.

  Unlocked parameter values can be changed in the drive after download.

Output pins

Every output pin has two pin names:

- *Internal name* is inside the block rectangle:
  - Internal name of a pin of a *normal* block describes the function of the pin.
  - Internal name of a pin of a *firmware* block is the parameter ID & name combination of its pin parameter.
- *External name* is used in all pin connections:
  - Default external name of an output pin of a *normal* block is `xxx(nnn)`, where `xxx` is the internal pin name and `nnn` is the block ID.
    Users can replace these names with their own pin names.
    Example of a default pin name: 

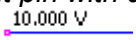
  - External name of an output pin of a *firmware block* is always the name of its pin parameter.

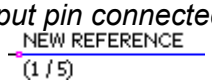
If the connected input and output pins are not on the same page, an arrowhead is displayed at the output pin: 

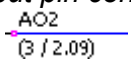
Input pins

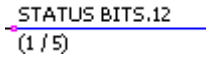
Every input pin is either set to a constant value or connected to an output pin (or to a bit of an output pin).

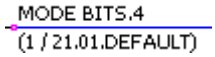
Below are input pin examples (all five possible cases are included):

- *Input pin with a constant value:*
  

- *Input pin connected to an output pin of a normal block or a CC instance:*
  

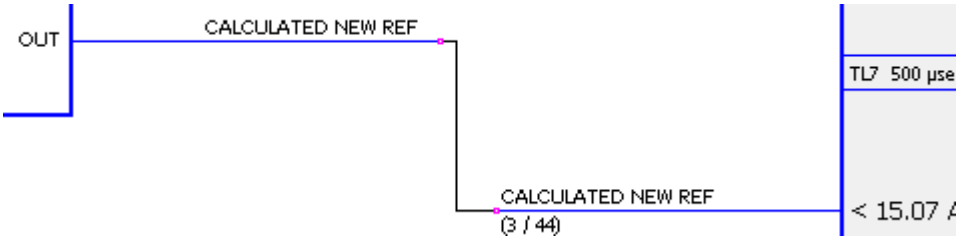
  The external name of the connected output pin is above the input pin.
  The page number and block ID of the output pin's block are below the input pin.
- *Input pin connected to an output pin of a firmware block:*
  

  The name of the pin parameter of the connected output pin is above the input pin.
  The page number of the output pin's block and the parameter ID of the output pin's pin parameter are below the input pin.
- *Input pin connected to a bit of an output pin of a normal block or a CC instance:*
  

  The external name and bit number of the connected output pin are above the input pin.
  The page number and block ID of the output pin's block are below the input pin.
- *Input pin connected to a bit of an output pin of a firmware block:*
  

  The name and bit number of the pin parameter of the connected output pin are above the input pin.
  The page number of the output pin's block, the parameter ID of the output pin's pin parameter and the name of the bit (or bit number if no bit name exists) are below the input pin.

A wire is automatically drawn between the connected input and output pins if these pins are on the same page.
Wire example:



Boolean input pins can be inverted. Inverted pins are marked with a little bubble, for example, 

User parameters

Drive firmware contains a fixed set of parameters.

Users can create parameters of their own by DriveSPC. User parameters are needed if the pin value setting and monitoring of programmed *normal* blocks is required by other means than DriveSPC (e.g., by panel).

There are two basic user parameter types:

- *Reference parameter*. These parameters are connected to input pins of normal blocks. Values of the connected input pins can now be set, e.g., by panel.
- *Signal parameter*. These parameters are connected to output pins of normal blocks. Values of the connected output pins can now be monitored, e.g., by panel.

The definition of a user parameter contains two steps:

- A parameter is created in the *Parameter Manager* window of DriveSPC (see Appendix 3)
- The created parameter is connected to a pin of a programmed normal block by an parameter block.

User alarms

Users can create alarms of their own by DriveSPC.

The definition of a user alarm contains two steps:

- An alarm is created in the *Alarm Manager* window of DriveSPC (see Appendix 4)
- The created alarm is connected to an output pin of a programmed block by a alarm block.

User faults

Users can create faults of their own by DriveSPC.

The definition of a user fault contains two steps:

- A fault is created in the *Fault Manager* window of DriveSPC (see Appendix 5)
- The created fault is connected to an output pin of a programmed block by a fault block.

Templates

Templates can be uploaded from drives.

A drive contains one or two templates:

- Empty template (this template always exists)
- Base solution (this template is optional)

Empty template

An empty template contains an empty solution program, i.e., instances of all firmware blocks, but no normal block instances.

Empty templates are used as the starting points of completely new solution programs.

Base solution

A drive can contain a base solution when it is delivered to a user.

A base solution is a complete solution program designed for a certain purpose.

Base solutions can be used as such, or they can be further enhanced by users.

Programming procedure

First programming steps

- If you are using a drive and you want to make a completely new program:
  Upload the empty template from the drive. All template information exists in the drive; no additional PC files are required.
- If you want to make a completely new program without a drive:
  Open an empty template file.
- If you are using a drive and you want to use its program (or base solution) as a base program:
  Upload the program (or base solution, if it exists) from the drive. All program information exists in the drive; no additional PC files are required.
- If you want to use an existing program as a base program:
  Open the existing program file.

Actual programming

Modify the program by:

- adding instances of normal blocks
- defining custom circuits and adding instances of them
- connecting pins
- . . .

Post-programming

- Save the modified solution program
- If you are using a drive and you want to run your new program in the drive:
  1. Connect to the drive (if you have not yet connected).
  2. Download the program to the drive. The drive is automatically restarted after the download and the program is running in the drive.
  3. If you want to monitor pin values on the running program, go to the *On-Line* mode and select the pins you want to monitor.

Use of DriveSPC

Chapter overview

This chapter describes the use of DriveSPC.

Starting DriveSPC

You start the DriveSPC program by clicking the **Start** button in the *Task Bar* (**Start > Programs > DriveWare > DriveSPC**, if you have used the default settings in the software installation).

If needed, several DriveSPC windows can be opened by starting the DriveSPC program several times.

User interface

The user interface of DriveSPC can be divided into two parts:

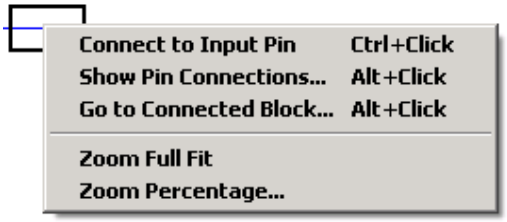
- *Menu commands*
- *Mouse clicks on the screen*

With *menu commands* you can, for example:

- open and save program files
- define, copy, save, read, and remove custom circuits
- connect to a drive
- upload and download programs
- start the *On-Line* mode

Click the *right mouse button* to display a context-sensitive menu. The commands of this menu depend on the program object, for example, a block or a pin you clicked on.

Menu example:



Select the desired command from the menu.

Click the *left mouse button* while holding down the **Ctrl**, **Alt**, or **Shift** key to quickly perform or start some often needed actions; see Appendix 1. In this way you can, for example:

- make connections between block pins
- move blocks on the screen
- monitor pin values in real time

If you press the **F1** key, help information about the currently active window, dialog box or menu item is displayed.

If you press the **F3** key, the description of the program item currently under the cursor (e.g. a block or a Structured Text keyword) is displayed.

Program pages

The solution program is divided into several pages. Every page has a user-specified number and a user-specified name.

There are tabs for all pages at the bottom of the screen. The number and optionally the name of the page are shown in the tabs. If the page names are not displayed in the tabs, they can be seen as a *ToolTip* when the cursor is moved on the tab.
If the page is empty, an asterisk is displayed in the corresponding tab before the page number.

Page selection

A program page can be selected in the following ways:

- Click the page tab of the desired page with the mouse
- Use the **Home**, **Page Up**, **Page Down**, and **End** keys of the keyboard

Page scrolling

The selected program page can be scrolled in the following ways:

- Use the scroll bars on the page
- Use the **Arrow** keys on the keyboard
- Use the mouse wheel (hold down the **Shift** key if you want to scroll horizontally)
- Drag the page with the mouse by holding down the left mouse button

Page zooming

The selected program page can be zoomed in the following ways:

- Click the right mouse button while holding down either the **Shift** or **Ctrl** key
- Use the mouse wheel (hold down the **Ctrl** key)

- Right-click anywhere on the page and select the desired zoom command. Possible zoom commands include:
  - **Zoom Percentage** opens a dialog box for the selection of the desired zoom percentage
  - **Zoom Actual Size** sets the zoom percentage to 100 %
  - **Zoom Full Fit** always displays the whole page regardless of the size of the page and size of the DriveSPC window
  - **Zoom Restore** restores the zoom percentage value that was in effect before the **Zoom Full Fit** command

First programming steps

Making a completely new program

- Using the empty template of a drive as a starting point*:*
  1. start DriveSPC
  2. upload the empty template from the drive by selecting **Drive > Upload Template from Drive**
- Using some other empty template file as a starting point*:*
  1. start DriveSPC
  2. open the empty template file by selecting **File > Open**

Making a new program based on an existing program

- Using the program (or base solution) of a drive as a base program*:*
  1. start DriveSPC
  2. upload the program from the drive by selecting **Drive > Upload Program from Drive** or base solution (if it exists) by selecting **Drive > Upload Template from Drive**
- Using some other program file as a base program*:*
  1. start DriveSPC
  2. open the program file by selecting **File > Open**

Actual programming (program modification)

After you have opened or uploaded the program or template, you can modify it as follows:

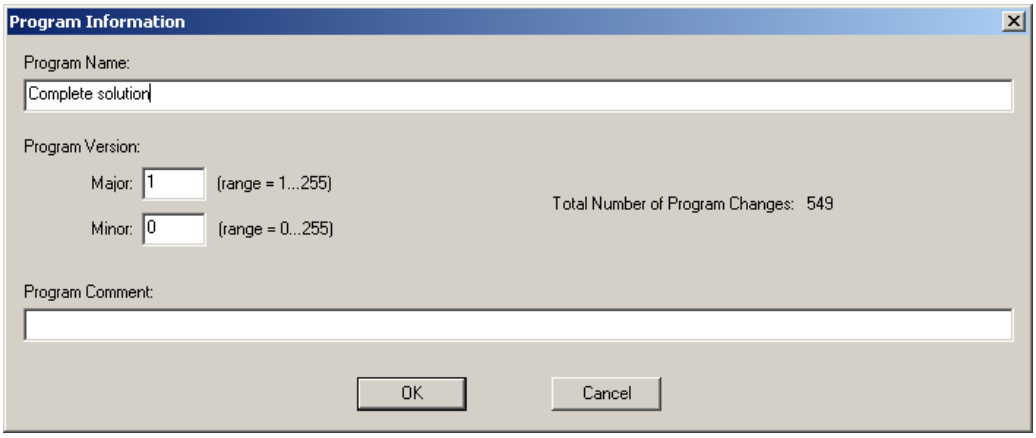
- Specify the name, version, and comment of your program
- Add or remove program pages
- Change the lengths, or execution intervals, of variable-length time levels
- Create, copy, save, read, and remove custom circuit definitions
- Add, modify, and remove instances of normal blocks and custom circuits
- Add, modify, and remove instances of comment blocks
- Move block and custom circuit instances on the screen
- Connect input and output pins
- Set values of input pins
- Invert input pins
- Define data items of array/structure pins
- Specify names of output pins
- Specify scaling factors and unit texts of output pins
- Create new user parameters, parameter groups, alarms, and faults
- Add and remove blocks for user parameters, alarms, and faults
- Specify the fields of the page header at the bottom of the pages

You can also perform the following programming-related actions:

- Display the execution sequence of block instances
- Go to the block connected to an input pin
- Show all connections of an output pin
- Go to a block connected to an output pin
- Highlight wires
- Zoom in and zoom out
- Show descriptions of blocks and pin parameters
- Protect program with user-specified passwords

Program information

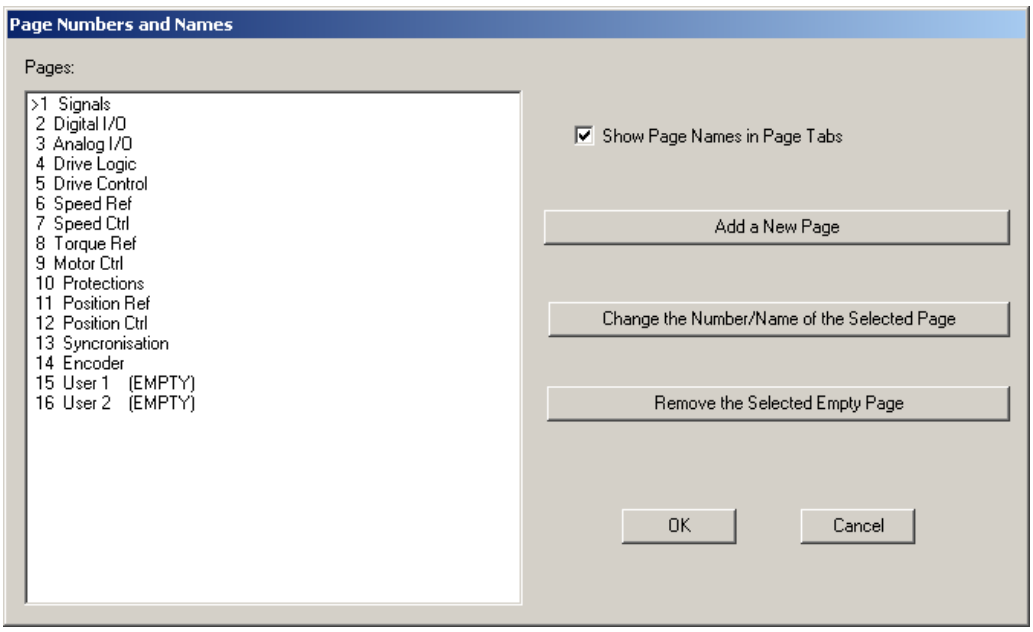
Select **Program > Program Information** to display a dialog box with the current name, version, and comment text of your solution program.



You can now change the name, version, and comment text.

Program pages

Select **Program > Program Pages** to display a dialog box with a list of the currently existing program pages.



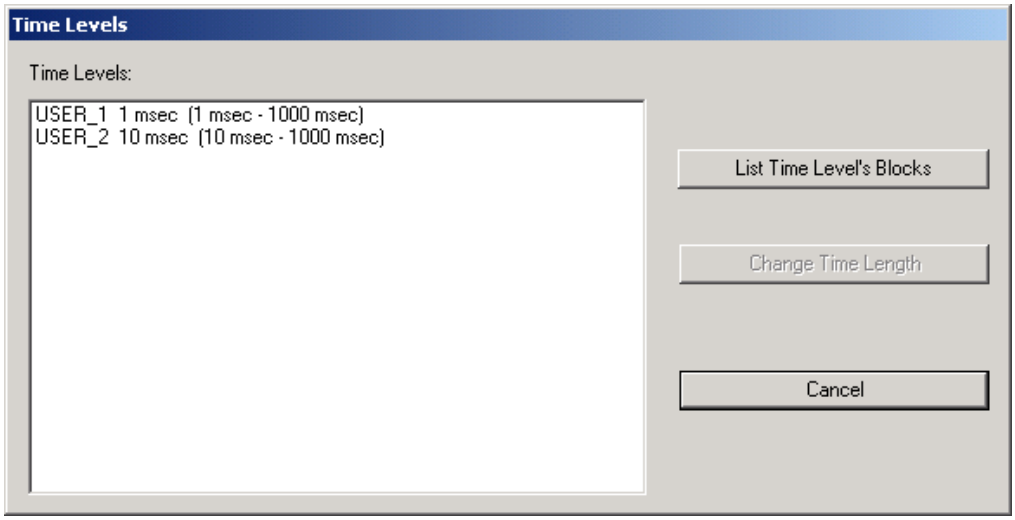
In the dialog box you can:

- add new pages and specify the numbers and names of new pages
- change numbers and names of the existing pages

- remove empty pages
- specify whether page names are displayed in page tabs. If page names are not displayed in the tabs, the name of the page can be seen as a *ToolTip* when the cursor is moved on the tab.

Time levels

Select **Program > Time Levels** to display a dialog box with a list of the time levels in the drive software.



If you want to see the list of the blocks in the selected time level, click **List Time Level's Blocks**.

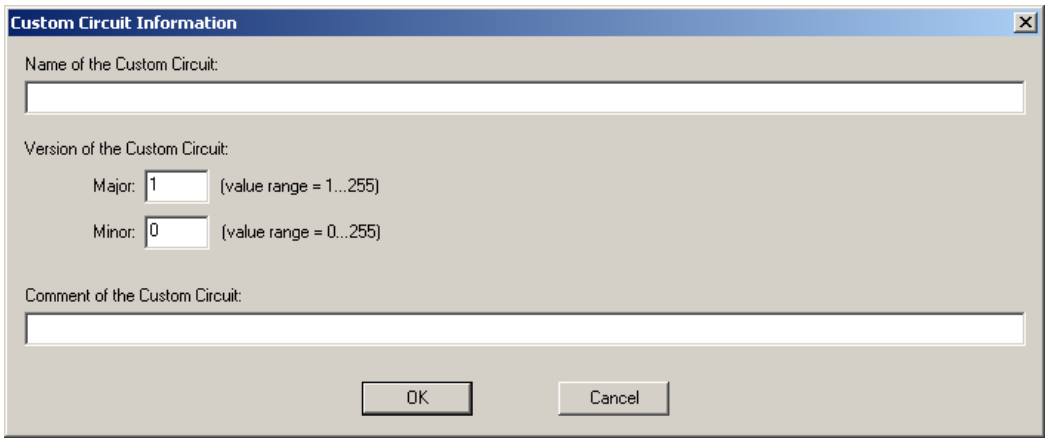
The allowable time range is displayed in the time level line.

You can now change the length of the selected variable-length time level.

Custom circuit definitions

Creating a new CC definition by using function blocks

Select **Program > New Custom Circuit (FB)**.
A dialog box is displayed:



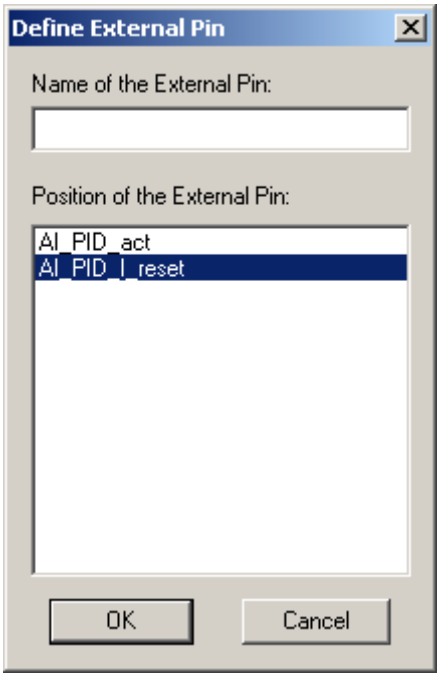
Specify the name, version, and comment text of the new CC definition.
The names of CC definitions always end with an asterisk ('*'). If you do not write this character at the end of the name, it will be appended automatically.

Click **OK**. The *CC definition window* opens. The window is initially empty and the text CUSTOM CIRCUIT is displayed at the top of the window.

You can now start to program the CC definition.
The definition programming is done in the same way as the actual solution programming in the solution program window.

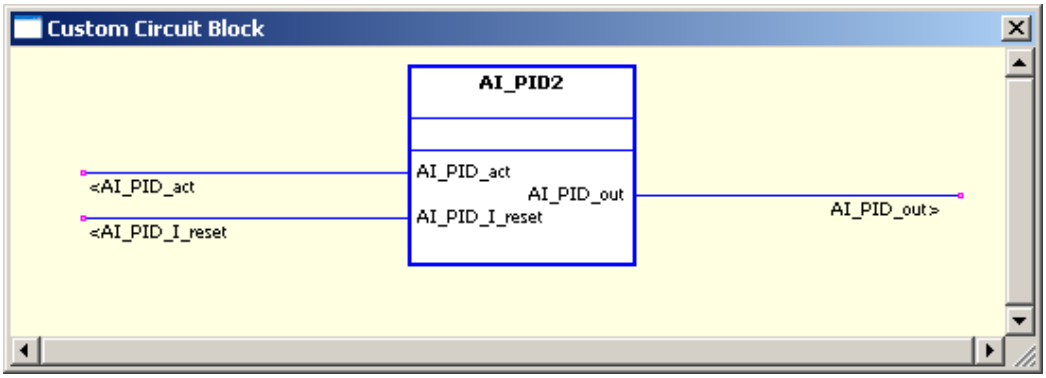
Every CC definition must include one or more **external pins**. These are input and/or output pins in the CC definition that will be visible in the instances of the CC definition. These pins make it possible to connect a CC instance to other blocks of the solution program.

To define an external pin, right-click on the input or output pin you want to be an external pin (if input pin, it must be unconnected and without any value) and select **Define External Pin**.
A dialog box is displayed:



The name you specify here will be seen inside the instance blocks of this CC definition in the solution programs.
In addition to the pin name, you have to specify the pin position among other external input or output pins. If no other external pins exist, number 1 is displayed in the pin position list.

All currently defined external pins can be seen by the **F5** key or by selecting **CustomCircuit > Custom Circuit Layout**.
The instance block of the CC definition will be displayed in the *CC block window*:

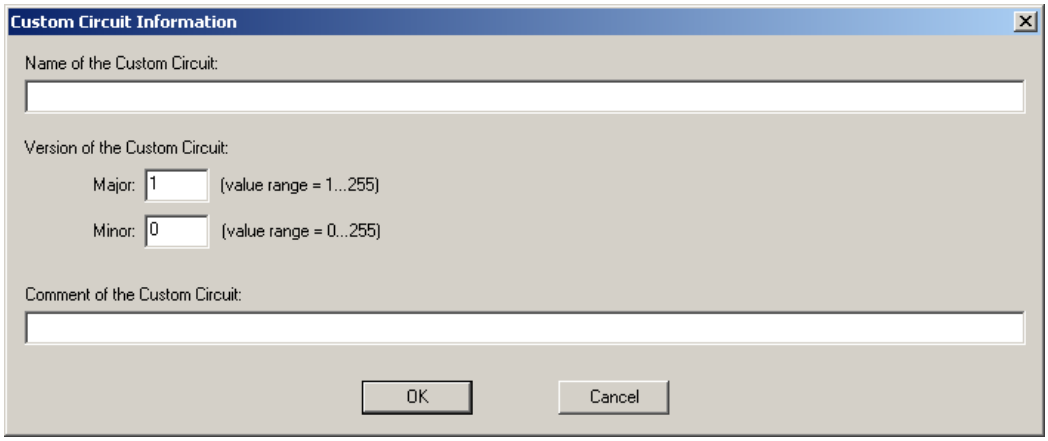


To find out the location of an external pin in the CC definition window, click on this pin in the CC block window.

When the CC definition is ready, select **File > Return to the Main Program**.
You will be asked if you want to preserve the CC definition you have made.

Creating a new CC definition by using the Structured Text language

Select **Program > New Custom Circuit (ST)**.
A dialog box is displayed:



Specify the name, version and comment text of the new CC definition.
The names of CC definitions always end with an asterisk ('*'). If you do not write this character at the end of the name, it will be appended automatically.

Click **OK**. The *CC definition window* opens. The window contains initially the following text:

```
CUSTOM_CIRCUIT

(* Add definitions of variables for external input pins (if any) here *)

(* Add definitions of variables for external output pins (if any) here *)

(* Add definitions of internal variables (if any) and
   definitions of function block instances (if any) here *)

(* Add executable program statements here *)

END_CUSTOM_CIRCUIT
```

You can now start to program the CC definition inside the **`CUSTOM_CIRCUIT ... END_CUSTOM_CIRCUIT`** lines.
The Structured Text language used in DriveSPC is described in *Appendix 6*.

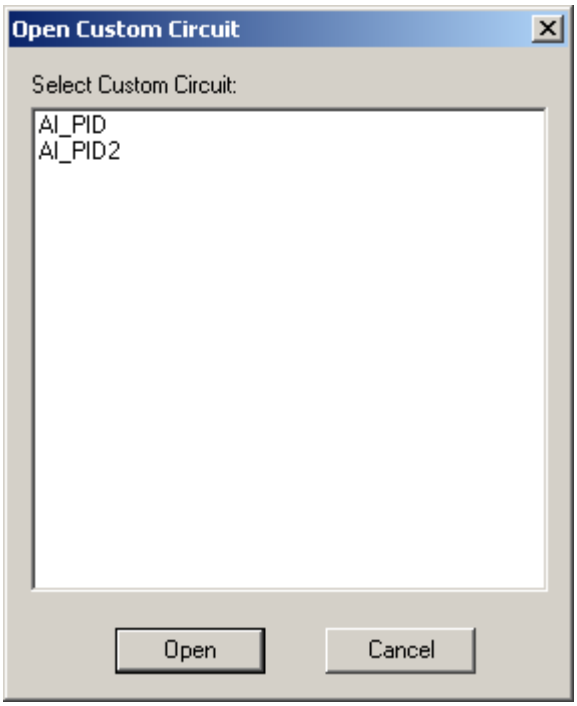
Every CC definition must include one or more **external pins**. These are input and/or output pins that will be visible in the instances of the CC definition. These pins make it possible to connect a CC instance to other blocks of the solution program.

External input pins are defined inside **`VAR_INPUT ... END_VAR`**.
External output pins are defined inside **`VAR_OUTPUT ... END_VAR`**.

When the CC definition is ready, select **File > Return to the Main Program**. You will be asked if you want to preserve the CC definition you have made.

Modifying an existing CC definition

Select **Program > Open Custom Circuit** to display a dialog box with a list of the currently defined custom circuits:



Select the CC definition you want to open and click **Open** or double-click on the desired CC.

The *CC definition window* opens with the CC definition you selected.

If needed, you can now modify the CC definition.

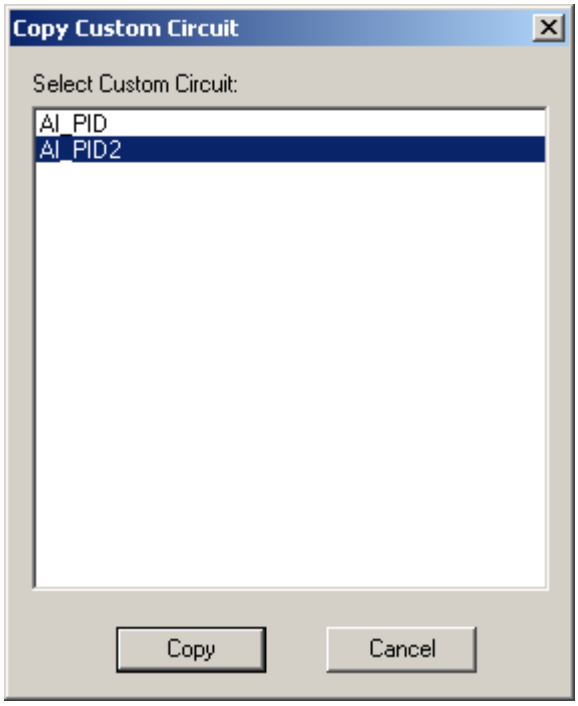
If you have used instances of this CC definition in your solution program, you can still make changes to this definition, as long as you do not change the definitions of external pins.

If you want to change the *data type* of an external pin in the CC definition and this pin is connected in one or more instances of this CC definition, you have to disconnect this pin in these instances before the pin change.

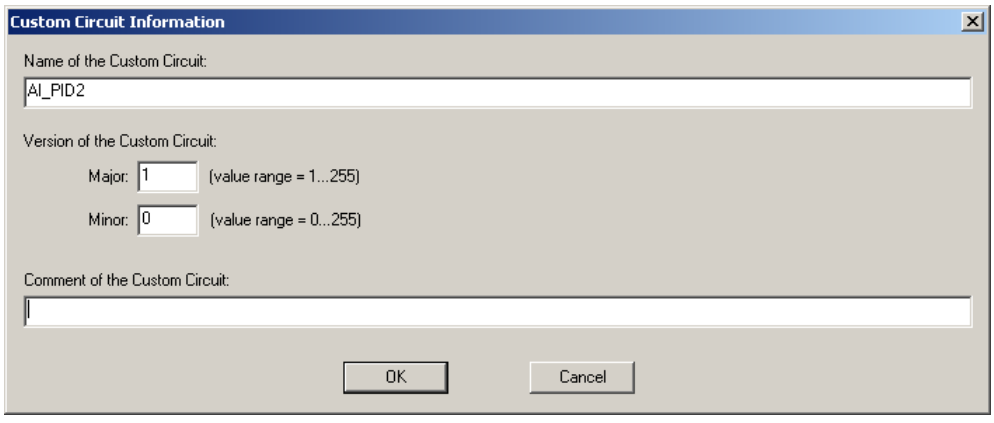
If you want to change the *number or input/output type* of external pins, you have to remove all instances of this CC definition from the solution program and add the instances again after you have changed the external pins in the CC definition.

Copying a CC definition

1. Select **Program > Copy Custom Circuit** to display a dialog box with a list of the currently defined custom circuits:



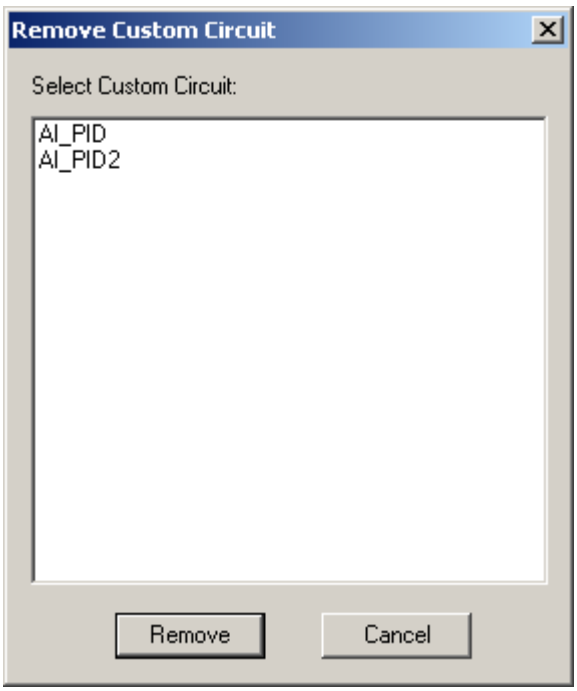
2. Select the CC definition you want to copy and click **Copy**, or double-click on the desired CC.
   A dialog box with the current name, version, and comment text of the CC definition to be copied is displayed:



3. Specify the new name, version. and comment text of the copy of the CC definition. Click **OK**.

Removing a CC definition

Select **Program > Remove Custom Circuit** to display a dialog box with a list of those currently defined custom circuits that are not used in the solution program:

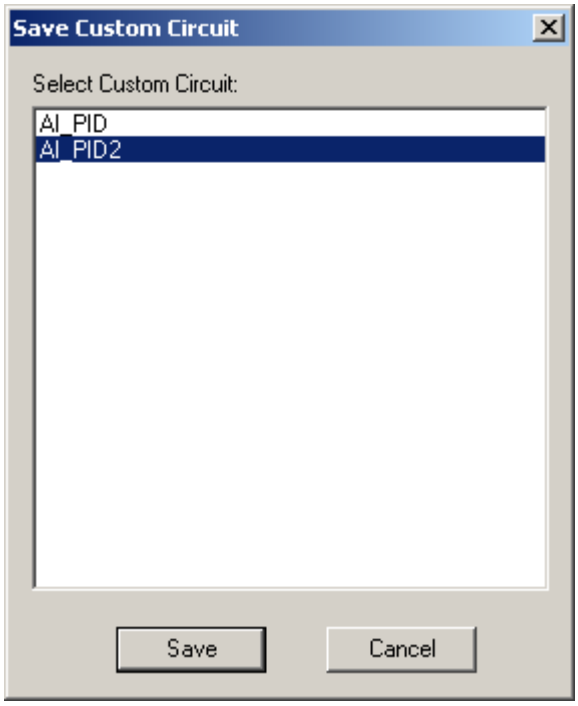


Select the CC definition you want to remove. Click **Remove**.

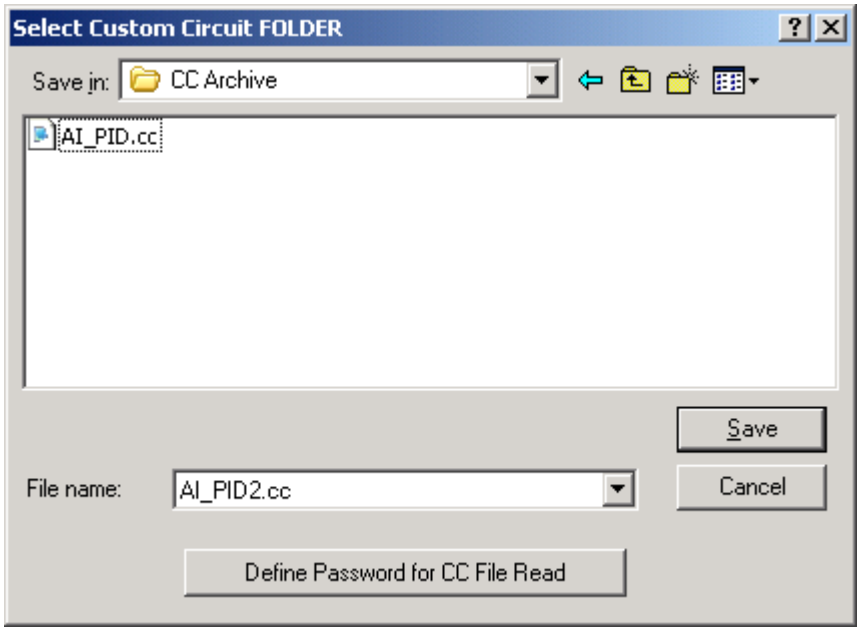
Saving a CC definition to a file

The save is needed *only if* you want to use a CC definition in other solution programs.

1. Select **Program > Save Custom Circuit to File** to display a dialog box with a list of the currently defined custom circuits:



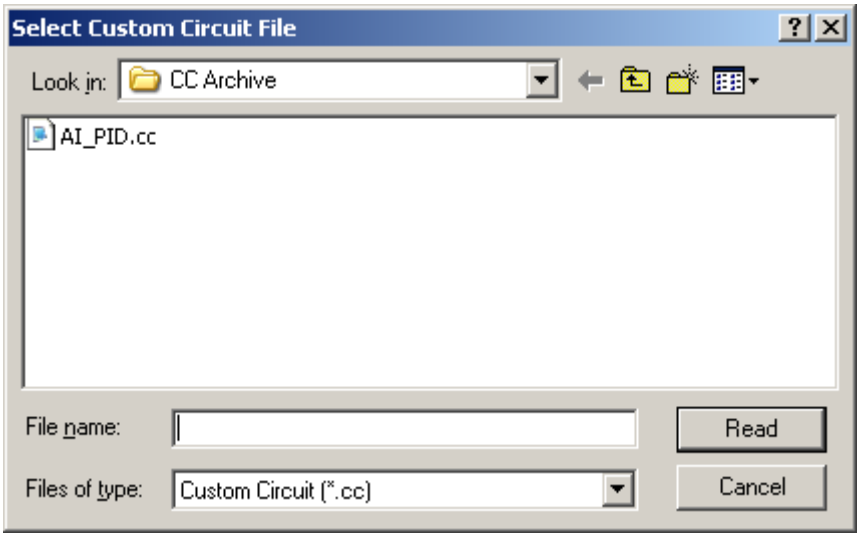
2. Select the CC definition you want to save and click **Save**, or double-click on the desired CC. Another dialog box is displayed:



3. If you want to require a password for reading the CC file, click **Define Password for CC File Read** and specify the password.
Select the desired CC archive folder and click **Save**.
Note: You cannot change the name of the CC file. Its name is always the name of the saved CC definition with extension "cc".

Reading a CC definition from a file

Select **Program > Read Custom Circuit from File**.
A dialog box is displayed:



Select the CC definition file you want to read. Click **Read**.

If a CC definition with the same name already exists in the solution program but instances of this CC definition are not used in the solution program, the existing CC definition in the solution program is always overwritten.

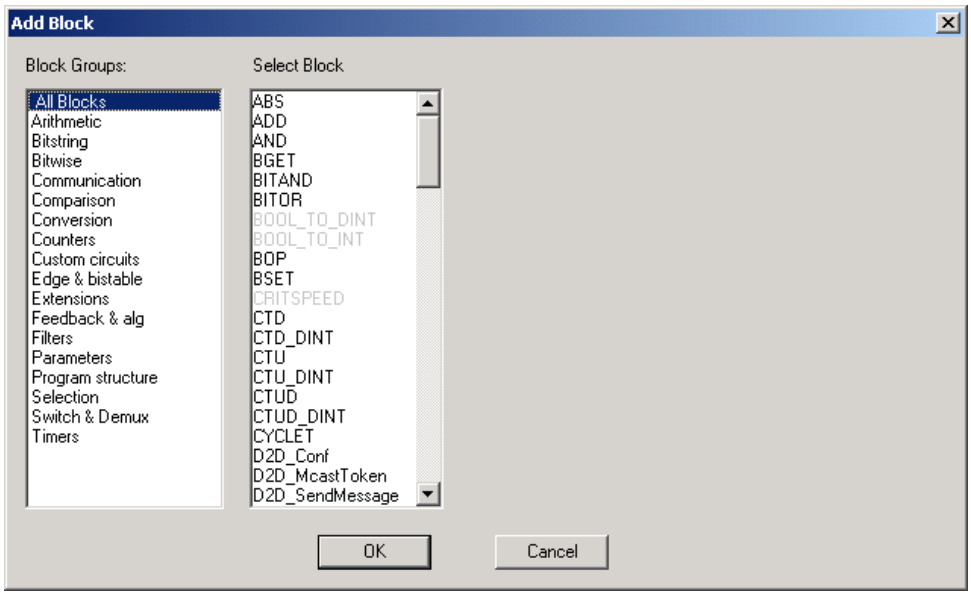
If instances of this CC definition are already used in the solution program and external pin definitions in the file are the same as in the existing CC definition, the existing CC definition in the solution program is always overwritten.

If instances of this CC definition are already used in the solution program but external pin definitions in the file are *not* the same as in the existing CC definition, the existing CC definition in the solution program remains unchanged. In this case, you have to remove all instances of this CC definition from the solution program and add the instances again after you have read the CC definition file.

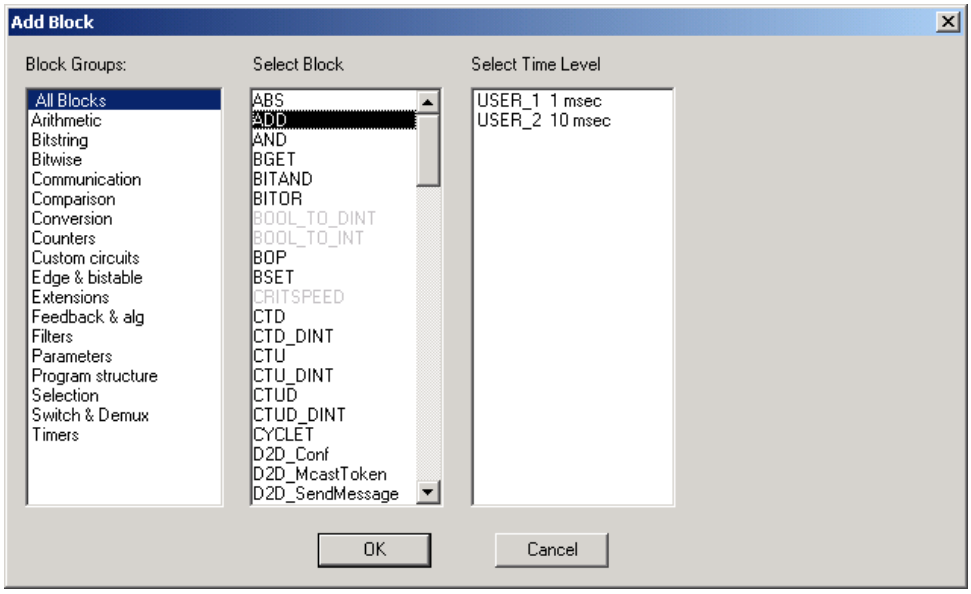
Block and CC instances

Adding a new instance of a normal block or a CC

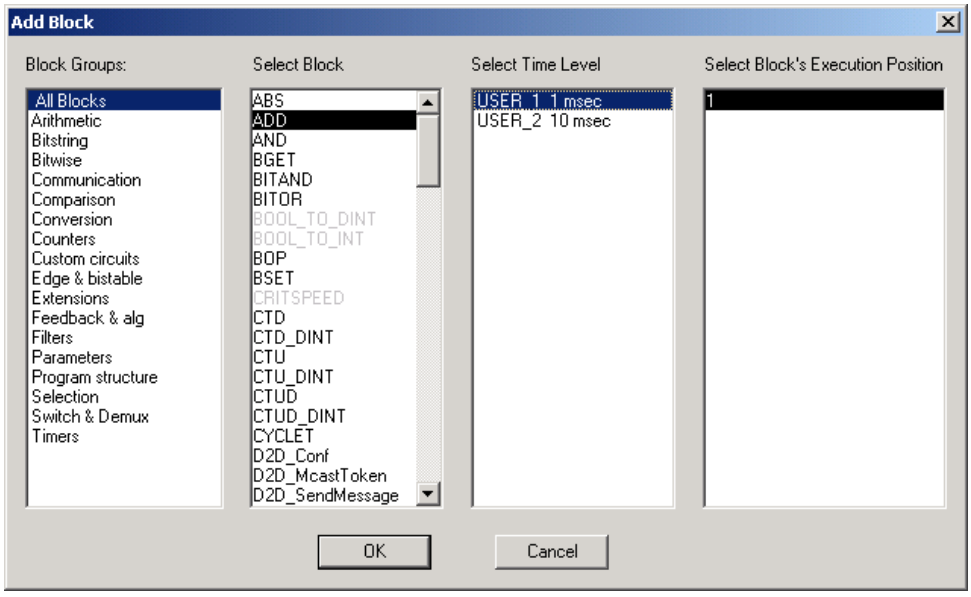
1. Right-click on an empty page location and select **Add Block**, or alternatively, left-click while holding down the **Ctrl** key.
   A dialog box with two lists is displayed. The first list contains block groups, of which the first group is always *All Blocks*. The second list contains the blocks and/or CCs of the selected block group. A grey block means that this block cannot be added because there is not enough empty space.



2. Select the block to be added.
   The third list containing the time levels is displayed.
   The time level list is not displayed if you are adding a block to a CC definition. All blocks in a CC definition are in the same unspecified time level. The actual time level is specified when you add an instance of the CC definition to your solution program.

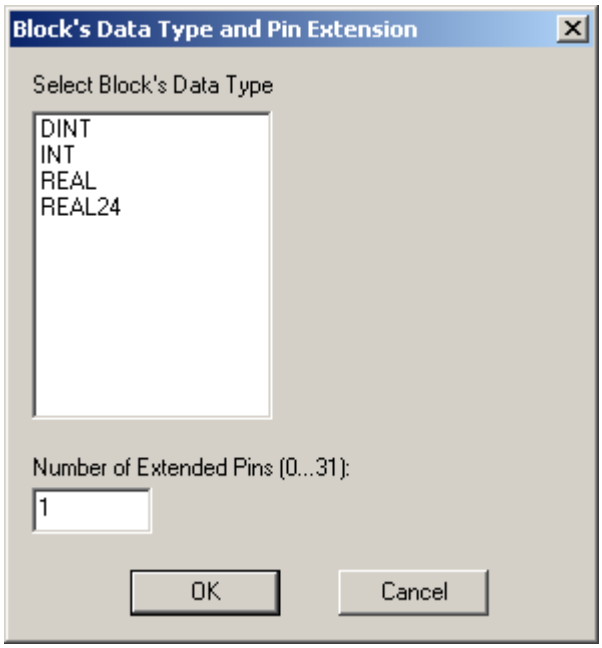


3. Select the time level where you want to put the new block instance (if the time level list exists) and the last list is displayed. The last list contains all blocks that are currently in the selected time level. If no blocks exist in the selected time level, number 1 is displayed in the list.



4. Select a block (or number 1 if there are no blocks) from the last list. Click **OK**. Grey blocks, if any, cannot be selected. The new block will be put *before* the selected block in the execution order of the selected time level. If you select the *last* block of the time level, you will be asked to specify whether the new block will be added before or after the selected block.
5. Finally, if the data type of the block to be added is user-selectable and/or if the block contains extensible pins, a dialog box will be displayed. If both data

type and extension size are needed, the dialog box is as follows:

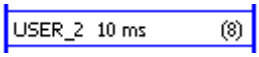


6. Select the block data type and/or the number of extended pins (i.e. the number of pin copies) of extensible pins. Click **OK**.

Displaying the execution sequence of blocks

Every block resides in some time level.

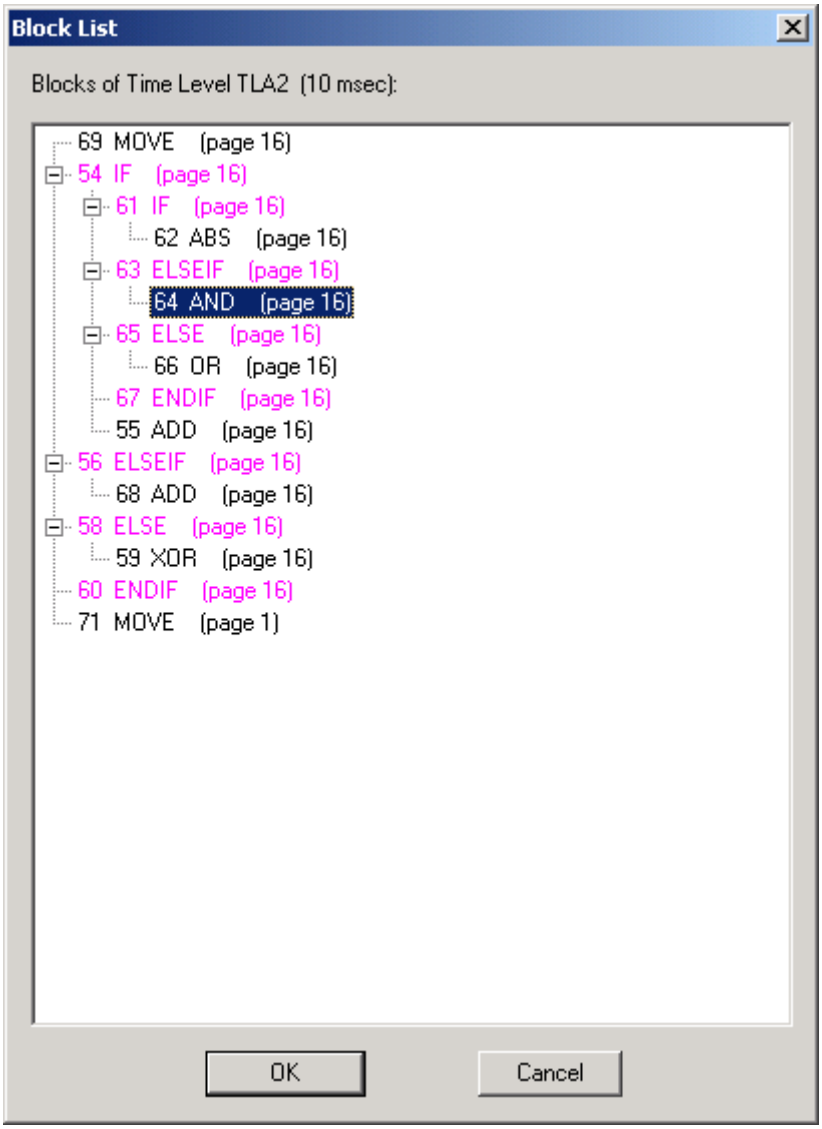
The execution position of a block in its time level is shown in parenthesis inside the block:



If you right-click on a block and select **Go to Next Block**, DriveSPC goes to the next block in the execution sequence.
If you now hold the **Ctrl** button down, you can continue going to the next blocks by pressing the **right arrow** key.

If you right-click on a block and select **Go to Previous Block**, DriveSPC goes to the previous block in the execution sequence.
If you now hold the **Ctrl** button down, you can continue going to the previous blocks by pressing the **left arrow** key.

If you right-click on a block and select **List Blocks**, a dialog box is displayed with a list of all blocks in the time level of the selected block:



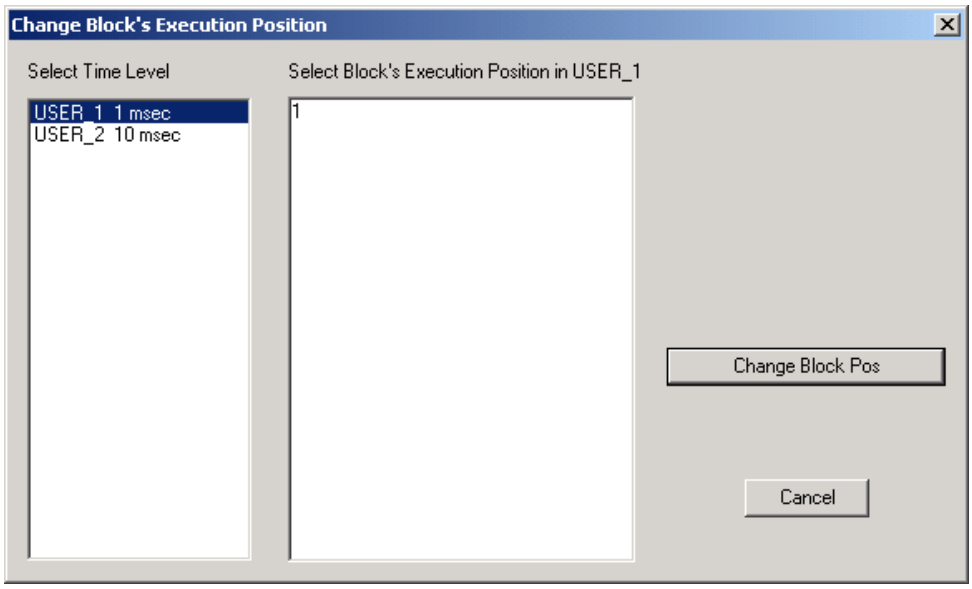
Magenta colour is used with conditional blocks IF, ELSEIF, ELSE, and ENDIF, if these blocks are used in the selected tíme level.
You can compress or expand conditional branches by clicking on their minus or plus buttons.

You can go to a block in the list by double-clicking on it in the list.

Changing the execution position of a normal block or CC instance

1. Right-click on the desired normal block or CC instance and select **Change Block Exec Pos**, or alternatively left-click while holding down the **Ctrl** key. If you are in the solution program window, a dialog box with two lists is displayed. The first list contains the time levels, where the current time level of the selected block is initially selected. The second list contains all blocks that are currently in the selected time level, or number 1 if no blocks exist in

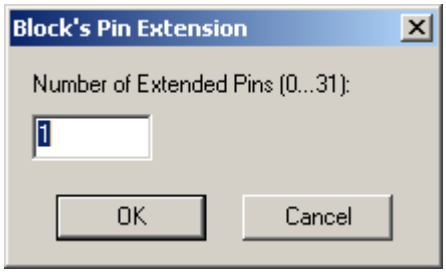
the selected time level.
If you are in the CC definition window, a dialog box with one list is displayed. This list contains all blocks that are currently in the CC definition.



2. Select the desired time level from the first list (if it exists) and the desired block from the second list. Any grey blocks cannot be selected.
3. Click **Change Block Pos**.
   The block whose execution position is to be changed will be put *before* the selected block in the execution order of the selected time level. If you select the *last* block of the time level, you are asked to specify whether the new block will be added before or after the selected block.

Adding/removing pins of a normal block that contains extensible pins

1. Right-click on the desired normal block and select **Add/Remove Pins**.
   A dialog box is displayed.



2. Change the number of extended pins (i.e. the number of pin copies). Click **OK**.

Moving a block or CC instance to another location

There are two move methods available: the Menu command method and the Quick method:

- *To move a block to another location by using the Menu command method:*
  1. Right-click on the desired block or CC instance, and select **Move Block**. A help message box is displayed. If you do not want the message box to be displayed during this DriveSPC session, select the check box.
  2. Right-click on the desired green location of the desired page, and select **Move Selected Block Here**.
- *To move a block to another location by using the Quick method:*
  1. Left-click on the desired block or CC instance while holding down the **Alt** key.
  2. Left-click on the desired green location of the desired page while holding down the **Alt** key.

Removing an instance of a normal block or a CC

Right-click on the desired normal block or CC instance and select **Remove Block**.

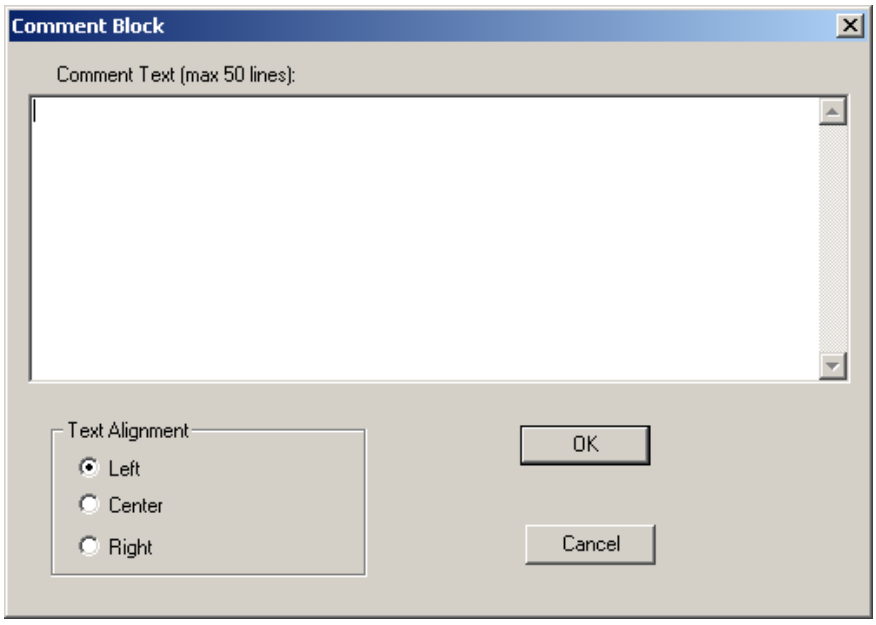
Displaying the description of a block or a CC

Move the cursor inside the rectangle of the desired block or CC instance and press the **F3** key.

Comments

Adding a new instance of the Comment block

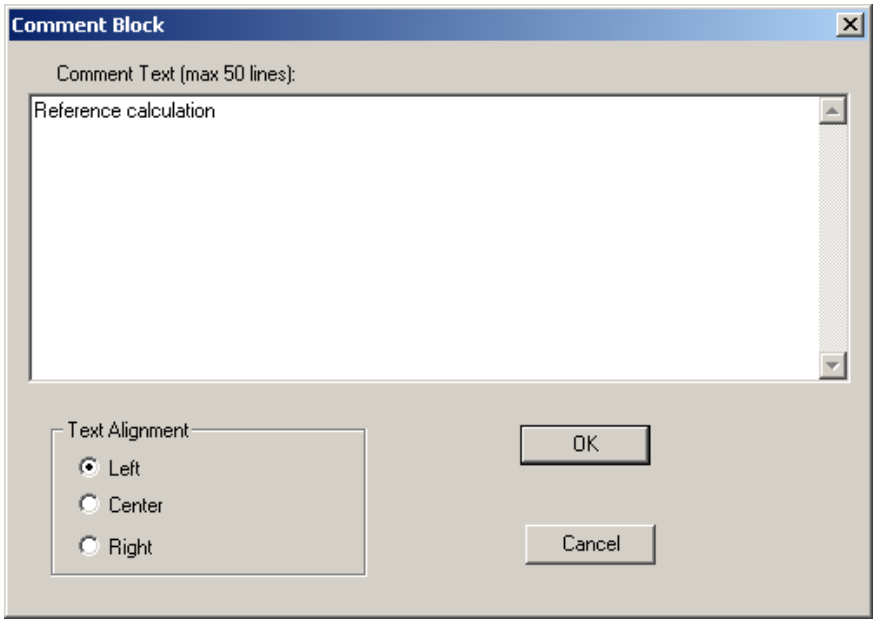
1. Right-click on an empty page location, and select **Add Comment Block**.
   A dialog box is displayed.



2. Enter the desired comment text. Select text alignment. Click **OK**.

Changing the text and/or text alignment of a comment block

1. Right-click on the desired comment block and select **Change Comment**.
   Alternatively, left-click while holding down the **Ctrl** key.
   A dialog box is displayed.



2. Change the comment text and/or change the text alignment. Click **OK**.

Moving a comment block to another location

There are two move methods available: the Menu command method and the Quick method:

- *To move a comment to another location by using the Menu command method:*
  1. Right-click on the desired comment block, and select **Move Block**. A help message box is displayed. If you do not want the message box to be displayed during this DriveSPC session, select the check box.
  2. Right-click on the desired green location of the desired page. Select **Move Selected Block Here**.
- *To move a comment to another location by using the Quick method:*
  1. Left-click on the desired comment block while holding down the **Alt** key.
  2. Left-click on the desired green location of the desired page while holding down the **Alt** key.

Removing an instance of the Comment block

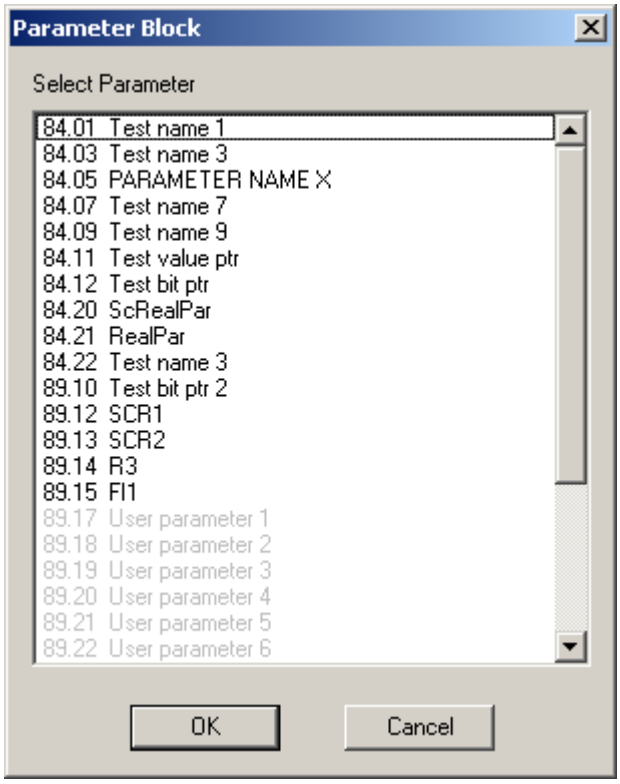
Right-click on the desired comment block. Select **Remove Block**.

Parameter blocks

Adding a new instance of a parameter block

Before parameter block adding is possible, one or more user parameters must have been defined using the Parameter Manager, see Appendix 3.

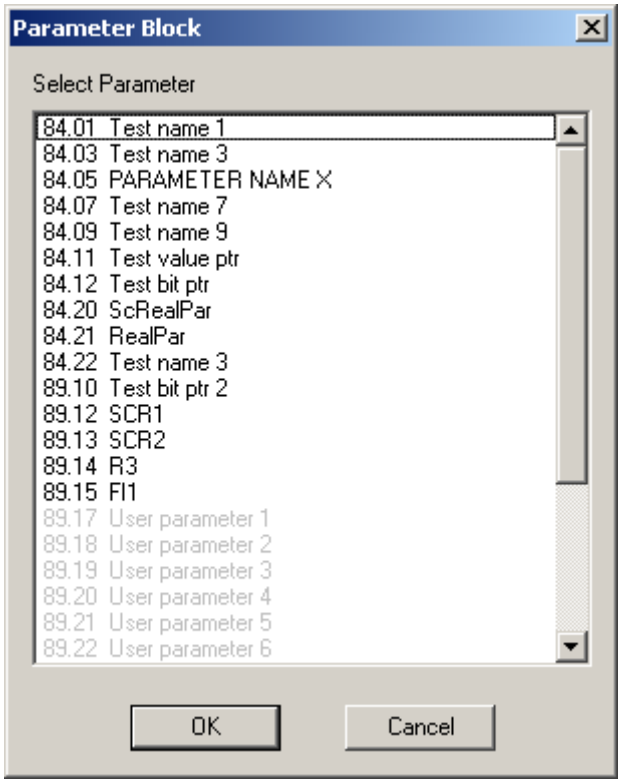
1. Right-click on an empty page location. Select **Add Parameter Block**. A dialog box is displayed.



2. Select the desired parameter. Click **OK**.

Changing the parameter of a parameter block

1. Right-click on the desired parameter block and select **Change Block's Parameter**. Alternatively, left-click while holding down the **Ctrl** key. A dialog box is displayed.



2. Select the desired parameter. Click **OK**.

Moving a parameter block to another location

There are two move methods available: the Menu command method and the Quick method:

- *To move a comment to another location by using the Menu command method:*
  1. Right-click on the desired parameter block. Select **Move Block**.
     A help message box is displayed. If you do not want the message box to be displayed during this DriveSPC session, select the check box.
  2. Right-click on the desired green location of the desired page. Select **Move Selected Block Here**.
- *To move a comment to another location by using the Quick method:*
  1. Left-click on the desired parameter block while holding down the **Alt** key.
  2. Left-click on the desired green location of the desired page while holding down the **Alt** key.

Removing an instance of a parameter block

Right-click on the desired parameter block and select **Remove Block**.

Displaying the description of the parameter of a parameter block

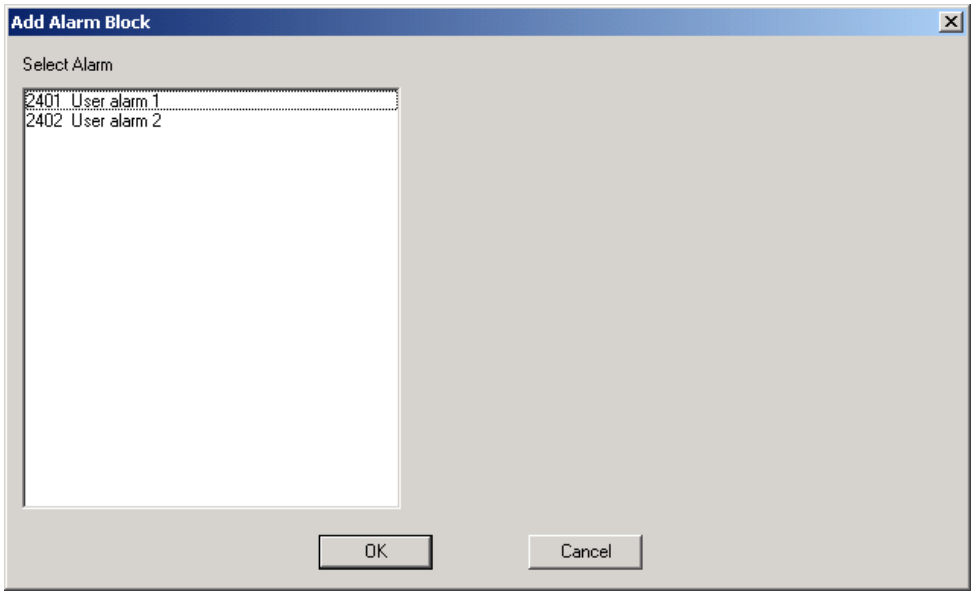
Move the cursor inside the rectangle of the desired parameter block and press the **F3** key.

Alarm blocks

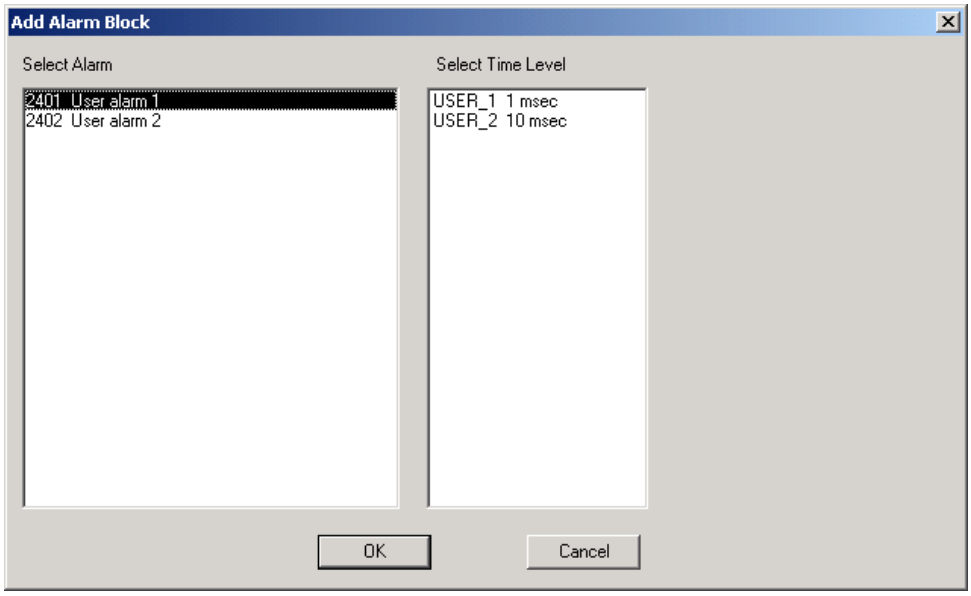
Adding a new instance of an alarm block

Before alarm block adding is possible, one or more user alarms must have been defined using the Alarm Manager, see Appendix 4.

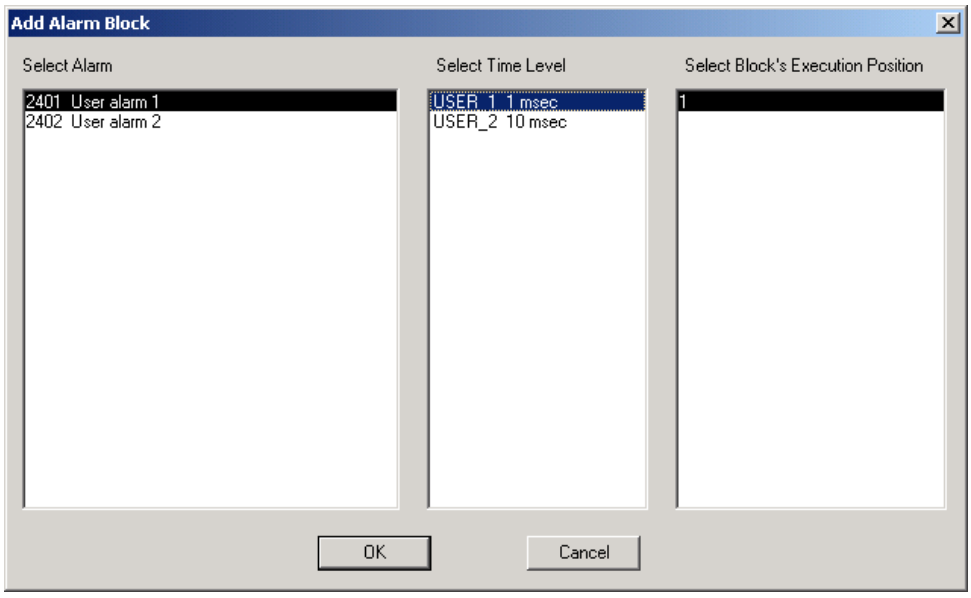
1. Right-click on an empty page location. Select **Add Alarm Block**.
   A dialog box is displayed.



2. Select the desired alarm.
   The second list containing the time levels is displayed.



3. Select the time level where you want to put the new block instance and the last list is displayed. The last list contains all blocks that are currently in the selected time level. If no blocks exist in the selected time level, number 1 is displayed in the list.



4. Select a block (or number 1 if there are no blocks) from the last list. Click **OK**. Any grey blocks cannot be selected. The new block will be put *before* the selected block in the execution order of the selected time level. If you select the *last* block of the time level, you are asked to specify whether the new block will be added before or after the selected block.

Displaying the execution sequence of blocks

Changing the execution position of an alarm block

Moving an alarm block to another location

Removing an instance of an alarm block

See the chapter, **Block and CC instances**.

Displaying the description of the alarm of an alarm block

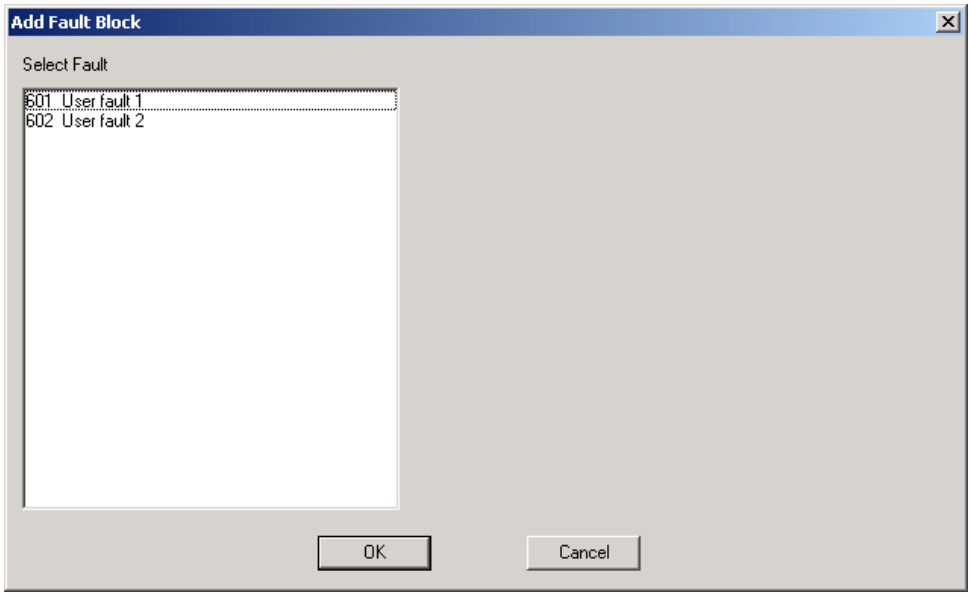
Move the cursor inside the rectangle of the desired alarm block and press the **F3** key.

Fault blocks

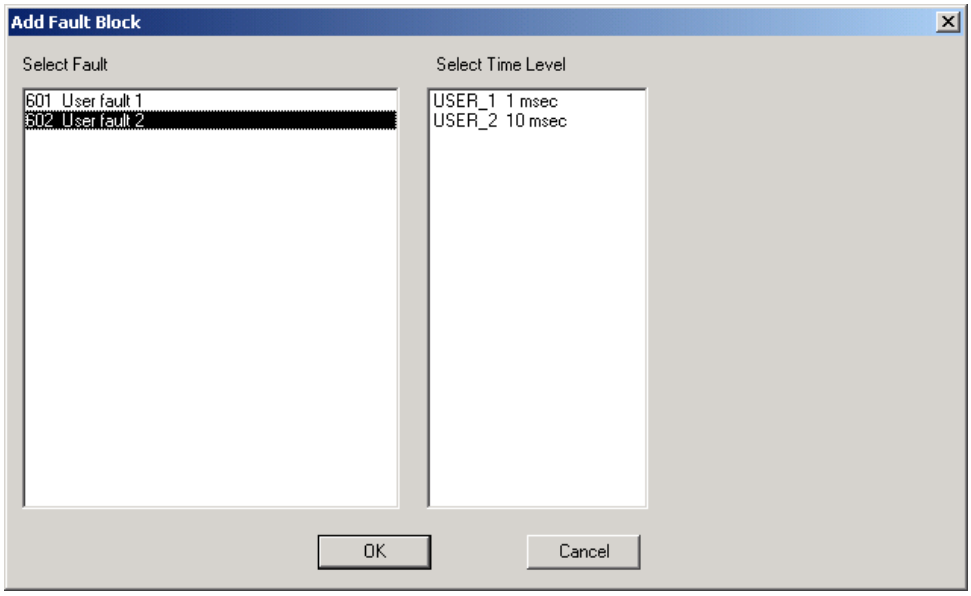
Adding a new instance of a fault block

Before fault block adding is possible, one or more user faults must have been defined using the Fault Manager, see Appendix 5.

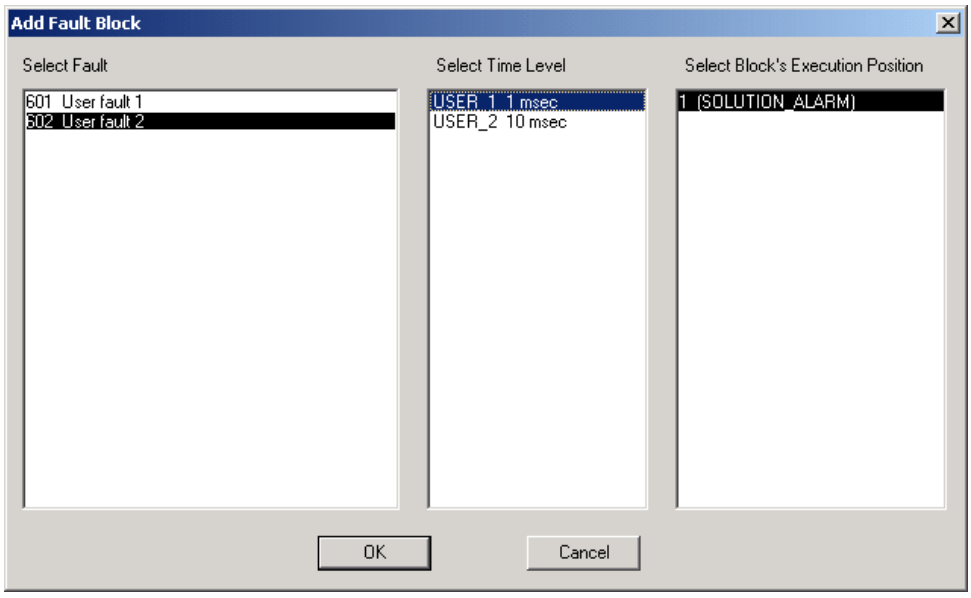
1. Right-click on an empty page location. Select **Add Fault Block**.
   A dialog box is displayed.



2. Select the desired fault.
   The second list containing the time levels is displayed.



3. Select the time level where you want to put the new block instance and the last list is displayed. The last list contains all blocks that are currently in the selected time level. If no blocks exist in the selected time level, number 1 is displayed in the list.



4. Select a block (or number 1 if there are no blocks) from the last list. Click **OK**. Any grey blocks cannot be selected. The new block will be put *before* the selected block in the execution order of the selected time level. If you select the *last* block of the time level, you are asked to specify whether the new block will be added before or after the selected block.

Displaying the execution sequence of blocks

Changing the execution position of an alarm block

Moving an alarm block to another location

Removing an instance of an alarm block

See the chapter, **Block and CC instances**.

Displaying the description of the fault of a fault block

Move the cursor inside the rectangle of the desired fault block and press the **F3** key.

Pins

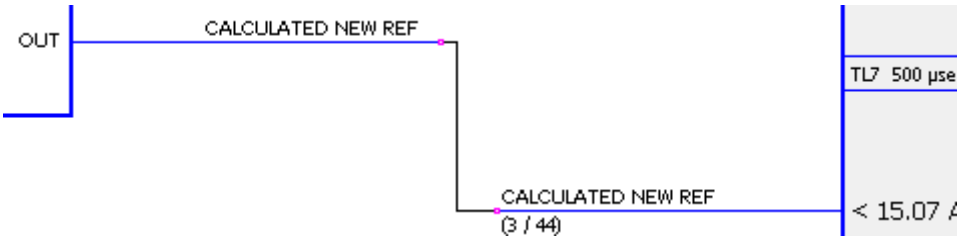
Connecting input and output pins

There are two pin connection methods available: the Menu command method and the Quick method:

- *To connect pins by using the Menu command method:*
  1. Right-click on the desired input pin and select **Connect to Output Pin** or right-click on the desired output pin and select **Connect to Input Pin**. A help message box is displayed. If you do not want the message box to be displayed during this DriveSPC session, select the check box.
  2. Right-click on the desired green pin and select **Connect to Selected Pin**.
- *To connect pins by using the Quick method:*
  1. Left-click on the desired input or output pin while holding down the **Ctrl** key.
  2. Left-click on the desired green pin while holding down the **Ctrl** key.

A wire is automatically drawn between the connected input and output pins if these pins are on the same page. If they are not on the same page, an arrowhead is displayed at the output pin: 

Wire example:



Connecting an input pin to a bit of an output pin

This bit connection is automatically used in the following two pin connection cases:

- The input pin is a *bit pointer* pin of a firmware block
- The data type of the input pin of a *normal* block is *Boolean* and the data type of the connected output pin is *other than Boolean*

There are two bit connection methods available, the Menu command method and the Quick method:

- *To connect a bit by using the Menu command method:*

1. Right-click on the desired input pin and select **Connect to Output Pin** (or **Connect to Bit of Output Pin**) or right-click on the desired output pin and select **Connect to Input Pin**.
   A help message box is displayed. If you do not want the message box to be displayed during this DriveSPC session, select the check box.
2. Right-click on the desired green pin and select **Connect to Selected Pin**.
3. A list of the bits of the selected output pin is displayed. Select the desired bit and click **OK**.

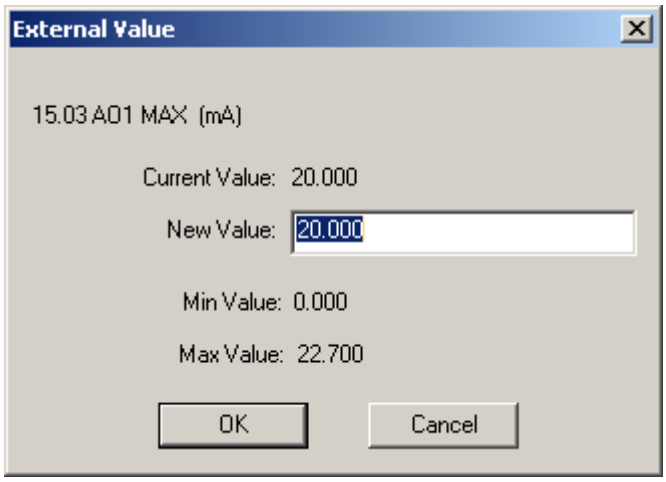
- *To connect a bit by using the Quick method:*

  1. Left-click on the desired input or output pin while holding down the **Ctrl** key.
  2. Left-click on the desired green pin while holding down the **Ctrl** key.
  3. A list of the bits of the selected output pin is displayed. Select the desired bit and click **OK**.

A wire is automatically drawn between the connected input and output pins if these pins are on the same page. If they are not on the same page, an arrowhead is displayed at the output pin.

Setting the value of an input pin

1. Double-click on the desired input pin or right-click on the desired input pin, and select **Set Value**.
   A dialog box is displayed. The content of the dialog box depends on the selected input pin. One possible dialog box is shown below:



2. Enter the desired new value or select the new value from the value list (if any). Click **OK**.

Setting the input pin of a firmware block to its default value

Right-click on the desired input pin of a firmware block and select **Set Default Value**.

Setting the input pin of a firmware block to drive value

Right-click on the desired input pin of a firmware block and select **Drive Value**.
Text `Drive value` is displayed in the pin.
When the program is downloaded, the current drive value of this parameter is preserved during the download operation.

Locking the value/connection of an input pin of a firmware block

Right-click on the desired input pin of a firmware block and select **Lock Value**.
When the program is downloaded, this value or pin connection is set to the drive parameter.

Unlocking the value/connection of an input pin of a firmware block

- Right-click on the desired input pin of a firmware block, and select **Unlock Value**.
  Unlocked value is enclosed with angle brackets, e.g. `[ 1.25 ]`.
  When the program is downloaded, the user can select one of the following two alternatives:
  - All unlocked values and pin connections are set to drive parameters.
  - The current drive values of all these parameter are preserved.

  Unlocked parameter values can be changed in the drive after download.

Removing the connection/value of an input pin of a normal block or CC instance

Right-click on the desired input pin of a normal block or CC instance and select **Disconnect** (or **Remove Value**).

Inverting an input (Boolean) pin of a normal block

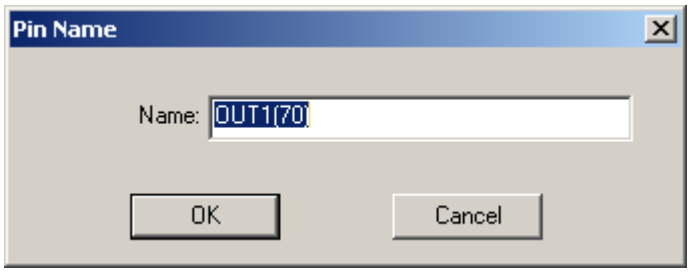
Right-click on the desired input pin of a normal block and select **Invert Pin**.
Inverted pin is marked with a little bubble, for example 

Defining the data items of an array/structure pin of a normal block or CC instance

There can be an array or a structure in an input or output pin of a normal block or CC instance.
The definition of these arrays and structures is described in Appendix 2.

Changing the name of an output pin of a normal block or CC instance

1. Right-click on the desired output pin of a normal block or CC instance, and select **Change Pin Name**.
   A dialog box is displayed.

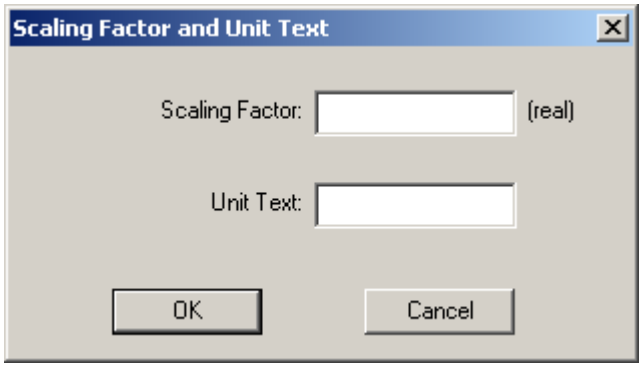


2. Enter the new unique name. Click **OK**.

Defining the scaling factor and unit text of an output pin of a normal block or CC instance

Scaling factors and unit texts are used in the *On-Line* mode. Scaling factors are real numbers, for example `2.5`.

1. Right-click on the desired output pin of a normal block or CC instance, and select **Define Scaling Factor and Unit Text**.
   A dialog box is displayed.



2. Enter the desired scaling factor and unit text. Click **OK**.

Saving the value of an output pin of a normal block or CC instance

If you want the value of an output pin of a non-firmware block to be retained during the power down of the drive, right-click on the desired output pin and select **Power Down Save On**.

Letter M will be displayed in this output pin.

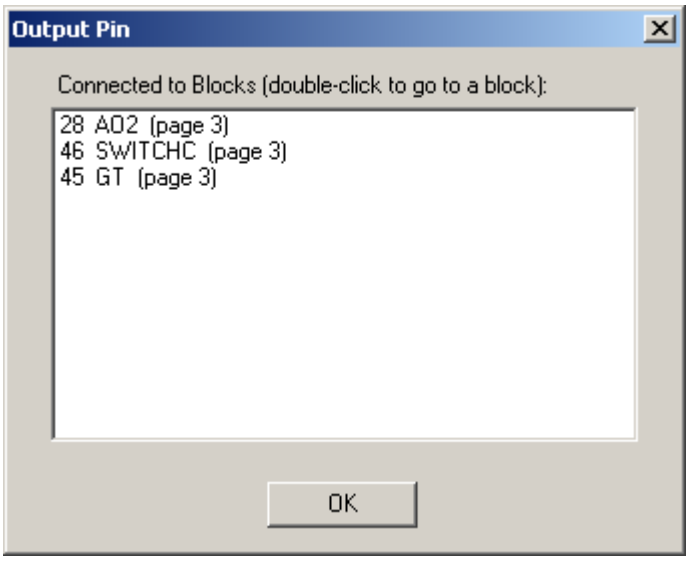
Note: Only a limited number of output pins can be saved.

Going to the block connected to an input pin

Right-click on the desired input pin and select **Go to Connected Block** or alternatively, left-click while holding down the **Alt** key.

Showing connections of an output pin

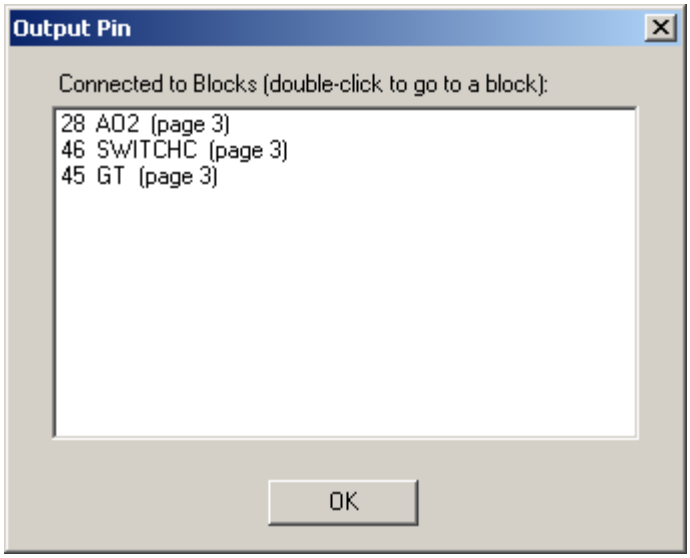
Double-click on the desired output pin or right-click on the desired output pin and select **Show Pin Connections** or alternatively, left-click while holding down the **Alt** key.
A dialog box is displayed with a list of the blocks whose input pins are connected to the output pin.



Going to a block connected to an output pin

1. Double-click on the desired output pin or right-click on the desired output pin and select the menu command **Go to Connected Block** or alternatively, left-click while holding down the **Alt** key.

A dialog box is displayed with a list that contains the blocks whose input pins are connected to this output pin.



2. Select the desired block and click **OK** or double-click on the desired block.

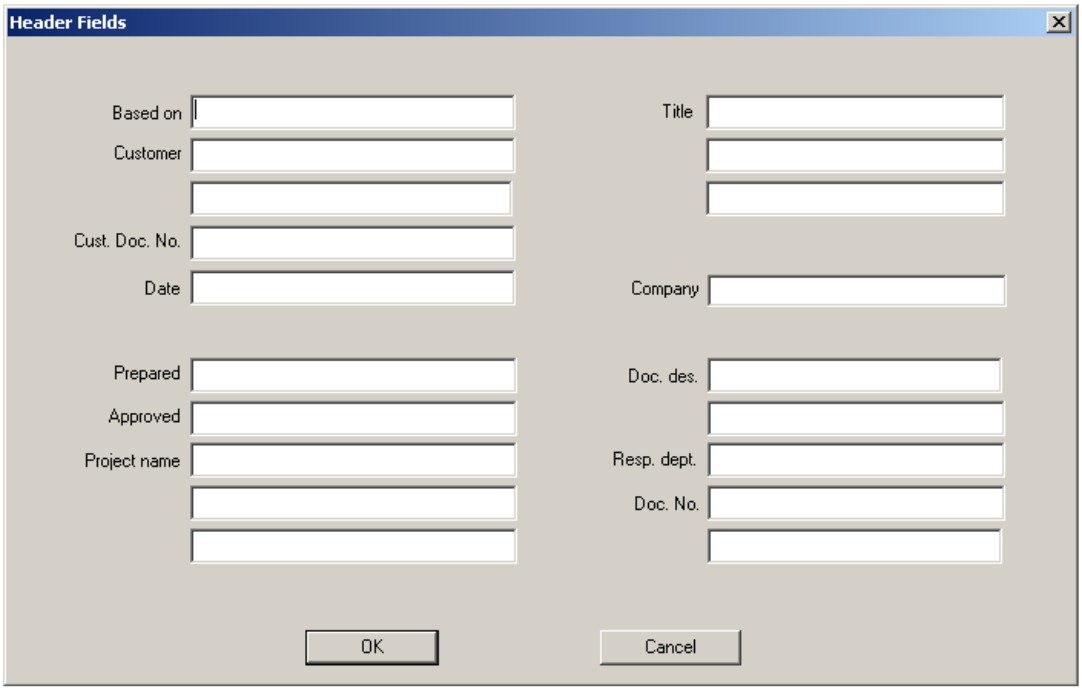
Displaying the description of a pin parameter of a firmware block

Move the cursor to the desired pin of a firmware block and press the **F3** key.

Page headers

All page headers at the bottom of the program pages contain the same field texts.

To change the text of one or more header fields, right-click on the page header on any page and select **Change Header Fields**.
A dialog box is displayed.

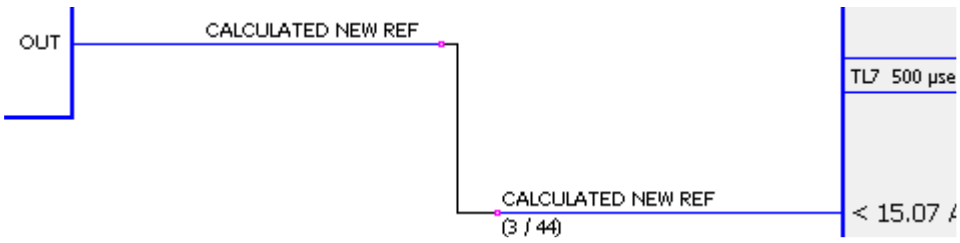


Enter the desired field texts and click **OK**.

Wires

A wire is automatically drawn between connected input and output pins if these pins are on the same page. If they are not on the same page, an arrowhead is displayed at the output pin: 

Wire example:



To highlight a wire (or a wire network if several input pins are connected to the same output pin), right-click on the desired wire and select **Highlight Wire**.
Alternatively left-click on a wire while holding down the **Alt** key.

Passwords

The following actions can be protected with user-specified passwords:

- Program open/upload
- Display of the opened/uploaded program. This password does not prevent the download of the opened/uploaded program to drive.
- Program download
- Display of a custom circuit definition included in the current program
- Reading of a custom circuit definition file to a program

Note: Do not forget your passwords!

A password for program open/upload is defined with the menu command **Program > Define Password for SP File Open**.

A password for the display of the opened/uploaded program is defined with the menu command **Program > Define Password for Program Display**.

A password for program download is defined with the menu command **Program > Define Password for Program Download**.
In order to enable the download of this program to a drive, the same password must be set to the drive with the menu command **Drive > Set Download Password to Drive** before the download.

A password for the display of a custom circuit definition included in the current program is defined with the menu command **CustomCircuit > Define Password for CC Display**.

A password for the reading of a custom circuit definition file to a program is defined with the button **Define Password for CC File Read** in the *Save Custom Circuit File* dialog box.

Post-programming

After you have made all program modifications, save the modified solution program by selecting **File > Save** or **File > Save As**.

To print one or more program pages, select **File > Print**.

To save the current program page on the screen to a disk file in the *Picture* (Enhanced Metafile) format, select **File > Save Horz Picture As**. The filename extension of this file must be `EMF`.
Picture files are used by, e.g., *MS Office* products. You can insert the saved program page file to a *Word* document by using Word's menu command **Insert > Picture > From File**.

You can rotate the current screen 90 degrees counter-clockwise and save it to a disk file by selecting **File > Save Vert Picture As**.

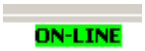
If you are using a drive and you want to run the new program in the drive:

1. Connect to the drive by selecting **Drive > Connect** (if you have not yet connected)
2. Download the program to the drive by selecting **Drive > Download Program to Drive**. The drive is automatically restarted after the download and the program is running in the drive
3. To monitor pin values of the running program, go to the *On-Line* mode by selecting **Drive > On-Line**.

On-Line mode

To change the operating mode of DriveSPC from the *Off-Line* mode to the *On-Line* Mode, select **Drive > On-Line**. The solution program of the drive is automatically uploaded, saved to a user-specified folder, and displayed on the screen.
Finally, the current values of all input pin parameters of all firmware blocks are read from the drive and displayed instead of the originally uploaded pin parameter values. If the read actual value of a pin parameter is not the same as its original value, this pin parameter value is displayed on a *yellow* background, for example 

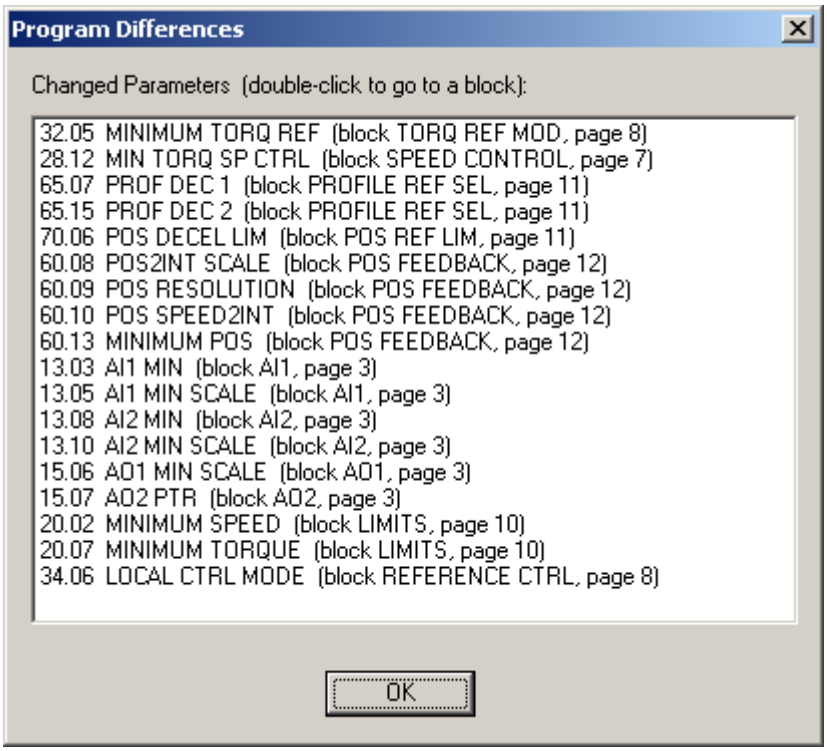
.

The text 

and the software load percentage of the drive are displayed at the top of the DriveSPC window.
The background colour of the text is red if there is a fault in the drive or yellow if there is a warning in the drive.

If you want to see the originally uploaded pin parameter values, press the **F4** key or select **Drive > Display Original Program**.
The text `ORIGINAL PROGRAM` is now displayed on the screen.
To view the actual pin parameter values again, press the **F4** key or select **Drive > Display Actual Program**.

To view a dialog box with a list of changed pin parameters, select **Drive > List Original/Actual Differences**.



To view a changed pin parameter, select the desired parameter and click **OK** or double-click on the parameter.

Changing values/connections of input pins of firmware blocks

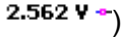
You can change values/connections of some input pins of *firmware* blocks as in the *Off-Line* mode. In the *On-Line* mode, these changes are automatically copied to the corresponding parameters of the connected drive.

Note that the changed *On-Line* pin parameter values are lost from *the program on the screen* when you go back to the *Off-Line* mode but they remain *in the drive*.
To change the actual *On-Line* pin parameter values/connections currently on the screen to programmed values of the drive:

1. Save the current program with actual values by selecting **File > Save**
2. Go back to the *Off-Line* mode by selecting **Drive > Off-Line**.
3. Open the saved program file by selecting **File > Open**
4. Download the program to the drive by selecting **Drive > Download Program to Drive**

Pin value monitoring

Actual values of one or more output pins, and input pins connected to output pins, can be monitored in two value formats:

- *External value* is the scaled value with unit text (black character colour, for example, ).
  External values can be used with pins of *norma*l blocks only if *scaling factors* and *unit texts* are defined for these pins.
- *Internal value* (magenta character colour, for example, ) is the value type actually used in the drive software

To monitor a pin value:

1. Right-click on the desired pin and select **Start Value Monitor** or alternatively, left-click on the pin while holding down the **Shift** key. If you select an output pin of a firmware block, the displayed value is initially the pin's external value (black). If you select an output pin of a normal block, the displayed value is initially the pin's internal value (magenta).
   - If no *scaling factor* and *unit text* are defined for an output pin of a *normal* block in the uploaded program and you want to monitor the external value of the pin, you have to define these before you start the monitoring of the pin. Right-click on the pin and select **Define Scaling Factor and Unit Text**. Note however that these *On-Line* definitions/changes are lost when you go back to the *Off-Line* mode. If you want non-volatile scaling factors and unit texts, specify these in the downloaded program.
2. If needed, change the value format by right-clicking on the pin and selecting **Monitor Internal Value** or **Monitor External Value**.

3. End the value monitoring of a pin by right-clicking on the pin and selecting **Stop Value Monitor** or alternatively, left-click on the pin while holding down the **Shift** key.

Monitoring of internal values of a custom circuit instance

Function block custom circuit

If you want to monitor values of internal pins of a custom circuit instance, move the cursor over the desired CC instance and press the **F3** key.

The CC definition window of this instance opens and you can now monitor actual pin values in this window in the same way as described in the previous chapter.

Structured Text custom circuit

If you want to monitor values of internal variables of a custom circuit instance, move the cursor over the desired CC instance and press the **F3** key.

The CC definition window of this instance opens and you can now click on the variables to be monitored.
The values are displayed in a separate Monitor window.

If you want to end the value monitoring of a variable, click on the variable again.

Return to the Off-Line mode

Go back to the *Off-Line* mode by selecting **Drive > Off-Line**.
The previously uploaded program is restored to the screen.

Menu commands (main program window)

Chapter overview

This chapter describes the menu commands of the main program window.

File menu

Open

This command reads and displays the contents of a previously saved program file.

The filename extension of program and template files is always `SP`.

The program name, program version, solution folder name, and filename of the opened file are displayed at the top of the DriveSPC window.

The opened program can be modified, printed, downloaded, and so on.

Save

This command saves the current program.

The solution folder name and filename are displayed at the top of the window.

Save As

This command saves the current program to a user-selected folder.

The solution folder name and filename will be updated at the top of the window.

Save Horz Picture As

This command saves the current program page to a disk file in the *Picture* (Enhanced Metafile) format.

The filename extension of this file is always `EMF`.

Picture files are used by, for example, *MS Office* products.
You can insert the saved program page file to a *Word* document by using Word's menu command **Insert > Picture > From File**.

Save Vert Picture As

This command rotates the current program page 90 degrees counter-clockwise and saves it to a disk file in the *Picture* (Enhanced Metafile) format.

The filename extension of this file is always `EMF`.

Picture files are used by for, example, *MS Office* products.
You can insert the saved program page file to a *Word* document by using Word's menu command **Insert > Picture > From File**.

Print

This command prints one or more program pages.

The page number of the currently visible page is displayed in the *Print* dialog box when you select **Print**.

Exit

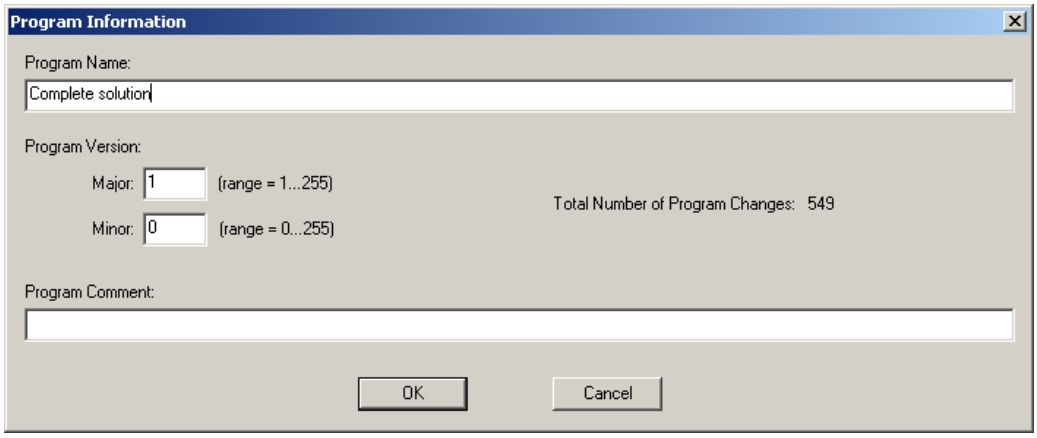
This command stops DriveSPC.

If you have changed this program but not saved it, you will be asked if you want to save it now.

Program menu

Program Information

This command displays a dialog box with the current name, version, and comment text of your solution program.

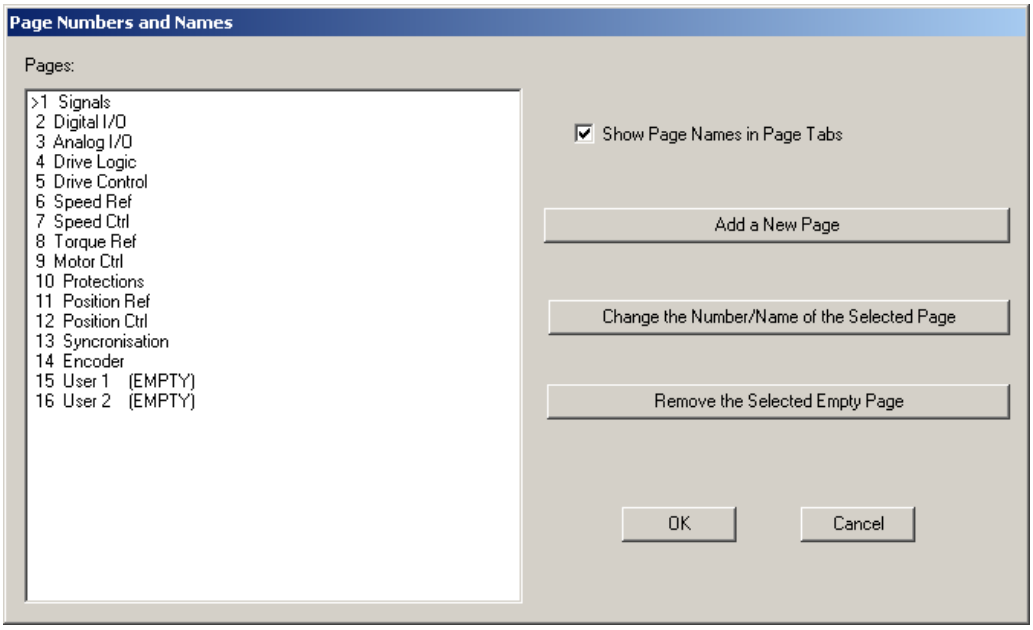


The program name and version are displayed at the top of the DriveSPC window, too.

If needed, change these items and click **OK**.

Program Pages

This command displays a dialog box with a list of the currently existing program pages.

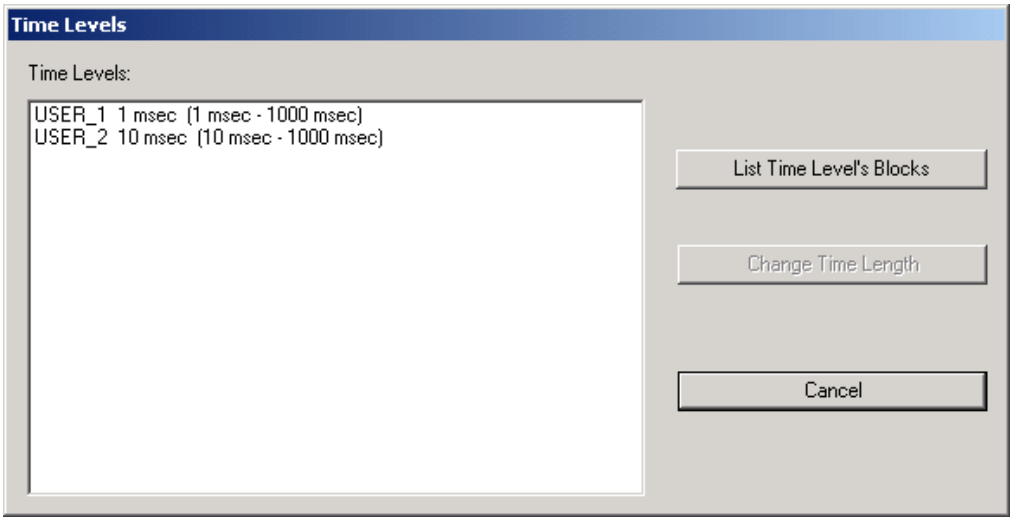


If needed, you can:

- Add new pages and specify numbers and names of new pages
- Change numbers and/or names of existing pages
- Remove empty pages
- Specify if page names are displayed in page tabs. If page names are not displayed in the tabs, the page name of the tab can be seen as a *ToolTip* when the cursor is moved onto the tab.

Time Levels

This command displays a dialog box with a list of the time levels in the drive software.



If you want to see the list of the blocks in the selected time level, click **List Time Level's Blocks**.

The allowable time range is displayed in the time level line.

To change the length of a variable-length time level, select the desired time level and click **Change Time Length**.

New Custom Circuit (FB)

This command initiates the creation of a new CC definition in the *CC definition window*. The new CC is defined by using function blocks.

New Custom Circuit (ST)

This command initiates the creation of a new CC definition in the *CC definition window*. The new CC is defined by using the Structured Text language, see Appendix 6.

Open Custom Circuit

This command opens a CC definition in the *CC definition window* for modification.

If you have used instances of the opened CC definition in your solution program, you can still make changes to this definition as long as you do not change definitions of external pins.

If you want to change the *data type* of an external pin in the opened CC definition and this pin is connected in one or more instances of this CC definition, you have to disconnect this pin in these instances before the pin change.

If you want to change the *number or input/output type* of external pins, you have to remove all instances of this CC definition from the solution program and add the instances again after you have changed the external pins in the CC definition.

Copy Custom Circuit

This command makes a copy of a CC definition.

Remove Custom Circuit

This command removes an unused CC definition.

Save Custom Circuit to File

This command saves a CC definition to a disk file.

The filename extension of CC definition files is always `CC`.

The save is needed *only if* you want to use this CC definition in other solution programs.

Read Custom Circuit from File

This command reads a CC definition from a previously saved disk file to the current solution program.

If a CC definition with the same name already exists in the solution program but instances of this CC definition are not used in the solution program, the existing CC definition in the solution program is always overwritten.

If instances of this CC definition are already used in the solution program and external pin definitions in the file are the same as in the existing CC definition, the existing CC definition in the solution program is always overwritten.

If instances of this CC definition are already used in the solution program but external pin definitions in the file are *not* the same as in the existing CC definition, the existing CC definition in the solution program remains unchanged.
In this case, you have to remove all instances of this CC definition from the solution program and add the instances again after you have read the CC definition file.

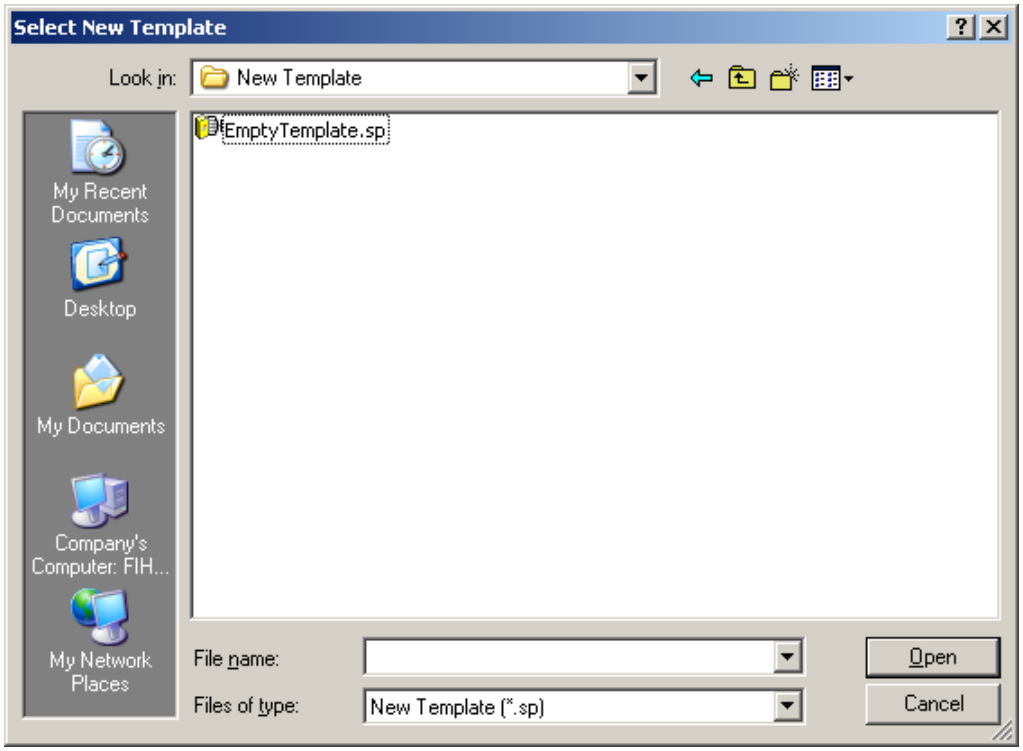
Change Program's Template

Every solution program is based on a program template.

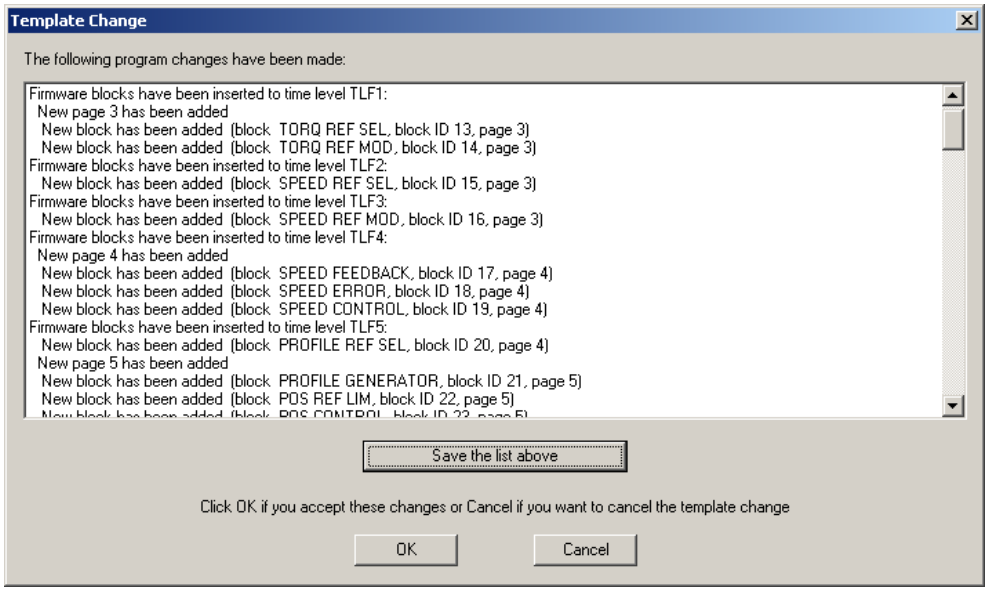
Templates are used as starting points of new solution programs. Templates contain the definitions of the blocks and other programming components that are in the drive software.
If a new drive software version is made where some programming components have changed, a new template for this new drive software can be uploaded from the new drive.

If you want to download your solution program to a drive, but the program is not based on the template that corresponds to the software of the drive, you have to change the template of your solution program as follows:

1. Upload template from your new drive by selecting **Drive > Upload Template from Drive** or, if the drive is not at hand now, obtain the template for the drive.
2. Open your solution program by selecting **File > Open**.
3. Select **Program > Change Program's Template**.
   A dialog box is displayed.



4. Select the new template and click **Open**.
5. The template change starts now and when it is ready, a dialog box is displayed:



6. All program changes made by DriveSPC are listed in this box. If you want to save this list to a disk file, click **Save the list above**.
7. If you accept these changes, click **OK**. The changed program will be saved.

8. The changed program may contain conversion errors. *Check the changed program carefully* and make any needed corrections before you download it to the drive.

Compare Programs

This command compares the solution program on the screen with a user-selected program.

List of program differences (if any) is displayed after the comparison.

Copy Page's Blocks

This command copies all non-firmware blocks of the current page to DriveSPC's internal memory.

Paste Copied Blocks

This command pastes the blocks copied with command **Program > Copy Page's Blocks** or **CustomCircuit > Copy Page's Blocks** to an empty page.

You can copy/paste blocks:

- From a program page to another program page
- From a program page to a page of a custom circuit definition
- From a page of a custom circuit definition to a program page
- From a page of a custom circuit definition to another page of a custom circuit definition
- From a solution program to another solution program

Parameter Manager

This command opens the Parameter Manager window (see Appendix 3).

Alarm Manager

This command opens the Alarm Manager window (see Appendix 4).

Fault Manager

This command opens the Fault Manager window (see Appendix 5).

Technology Library

In addition to the Standard block library that always exists, there are optional Technology block libraries. A Technology library contains one or more new blocks that are not in the Standard block library. Instances of these new blocks can be used in a solution program only if the same Technology block library is in the drive.

This command opens a dialog box where you can:

- Add a Technology library to your program. A program can include no more than one Technology library.
- See information about the blocks of program's Technology library
- Get a copy of the Technology library of your program
- Remove Technology library from your program

Define Password for SP File Open

If you want a password to be required when this program is opened or uploaded, select this command and specify the password.

Define Password for Program Display

If you want a password to be required for displaying this program when it has been opened or uploaded, select this command and specify the password.

This password does not prevent the downloading of the opened/uploaded program to a drive.

Define Password for Program Download

If you want a password to be required for downloading this program, select this command and specify the ID/password combination.
The ID must be an integer value in the range 1...65535.
The password must be an integer value in the range 1...4294967295.

In order to enable the download of this program to a drive, the same ID/password combination must be set to the drive with the menu command **Drive > Set Download Password to Drive** before the download.

Drive menu

Connect

If there is a drive connected to your PC hardware, connect DriveSPC to the drive by selecting **Drive > Connect**.
If there is more than one drive connected to the PC hardware, you select the drive from the list of available drives. If wanted, a separate DriveSPC window can be opened for each drive.

The identifier of the connected drive is displayed at the top of the DriveSPC window.

Disconnect

If there is more than one drive connected to your PC and you want to change the connection to another drive, disconnect the currently connected drive by selecting **Drive > Disconnect** and make a connection to another drive by selecting **Drive > Connect**.

Upload Program from Drive

This command uploads the previously downloaded solution program (if any) from the connected drive, saves it to a user-specified folder and displays it on the screen.

All program information exists in the drive; no additional PC files are required.

The program name, program version, solution folder name, and filename of the uploaded program are displayed at the top of the DriveSPC window.

The uploaded program can be modified, printed, downloaded, and so on.

Upload Template from Drive

If there is a base solution in the drive, a dialog box is displayed and you can select the template to be uploaded: either *empty template* or *base solution*.
Otherwise, the empty template of the drive is always uploaded.

The uploaded template is saved to a user-specified empty solution folder and displayed on the screen.

All template information exists in the drive; no additional PC files are required.

The program name, program version, solution folder name, and filename of the uploaded template are displayed at the top of the DriveSPC window.

The uploaded template can be modified, printed, downloaded, and so on.

Download Program to Drive

This command downloads the current solution program to the connected drive.

The drive is always automatically restarted after the download. The downloaded program is running in the drive after the restart.

Set Download Password to Drive

This command sets the user-specified ID/password combination to the connected drive for the download of a solution program that is protected with the same ID/password combination.
The ID must be an integer value in the range 1...65535.
The password must be an integer value in the range 1...4294967295.

The set ID value is displayed in the Properties window of DriveStudio. Its name is `APPL_LICENCE`.

Remove Program from Drive

This command removes the previously downloaded solution program from the connected drive.

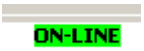
On-Line

This command changes the operating mode of DriveSPC from the *Off-Line* mode to the *On-Line* mode.

When the operating mode is changed to the *On-Line* mode, the solution program of the drive is automatically uploaded, saved to a user-specified folder and displayed on the screen.

Finally, the current values of all input pin parameters of all firmware blocks are read from the drive and displayed instead of the originally-uploaded pin parameter values.
If the read actual value of a pin parameter is not the same as its original value, it is displayed on a *yellow* background, for example 

.

The text 

and the software load percentage of the drive are displayed at the top of the DriveSPC window.
The background colour of the text is red if there is a fault in the drive, or yellow if there is a warning in the drive.

Off-Line

This command changes the operating mode of DriveSPC from the *On-Line* mode to the *Off-Line* mode.

The previously uploaded program is restored to the screen.

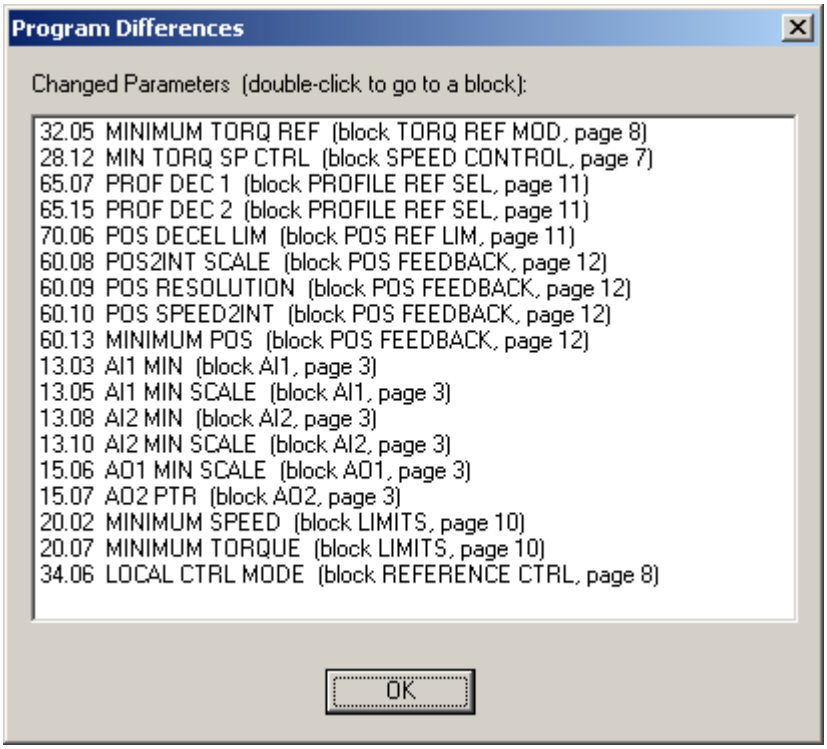
Display Original (or Actual) Program

This command (or the **F4** key) displays the solution program either with the originally uploaded pin parameter values or with the actual pin parameter values that were read from the drive.

The text `ORIGINAL PROGRAM` is displayed on the screen if the originally uploaded pin parameter values are currently on the screen.

List Original/Actual Differences

This command displays a dialog box with a list of changed pin parameters (the originally-uploaded pin parameter values of these changed parameters are not the same as the actual pin parameter values that were read from the drive).



If you want to see a changed pin parameter, select the desired parameter and click **OK** or double-click on the parameter.

Help menu

User Interface

This command displays the main index of the *DriveSPC Help*.

If you press the **F1** key, help information about the currently active window, dialog box or menu item is displayed.

Block/Parameter Information

This command opens the Firmware Manual of the drive in a separate PDF window.

If the cursor is on a firmware block, on a normal block, or on a pin parameter when you press the **F3** key, the description of this block or parameter is displayed in a separate PDF window.

If the cursor is on a CC instance when you press the **F3** key, the CC definition window with the definition of the CC instance is displayed.

If the cursor is on a parameter block when you press the **F3** key, the description of block's parameter is displayed.

If the cursor is on an alarm block when you press the **F3** key, the description of block's alarm is displayed.

If the cursor is on a fault block when you press the **F3** key, the description of block's fault is displayed.

About DriveSPC

This command displays the DriveSPC information box.

Menu commands (CC window, function blocks)

Chapter overview

This chapter describes the menu commands of the custom circuit (function blocks) definition window.

File menu

Save Horz Picture As

This command saves the current CC definition page to a disk file in the *Picture* (Enhanced Metafile) format.

The filename extension of this file is always `EMF`.

Picture files are used by, for example, *MS Office* products.
You can insert the saved program page file to a *Word* document by using Word's menu command **Insert > Picture > From File**.

Save Vert Picture As

This command rotates the current CC definition page 90 degrees counter-clockwise and saves it to a disk file in the *Picture* (Enhanced Metafile) format.

The filename extension of this file is always `EMF`.

Picture files are used by, for example, *MS Office* products.
You can insert the saved program page file to a *Word* document by using Word's menu command **Insert > Picture > From File**.

Print

This command prints one or more CC definition pages.

The page number of the currently visible page is displayed in the *Print* dialog box when you select **Print**.

Return to the Main Program

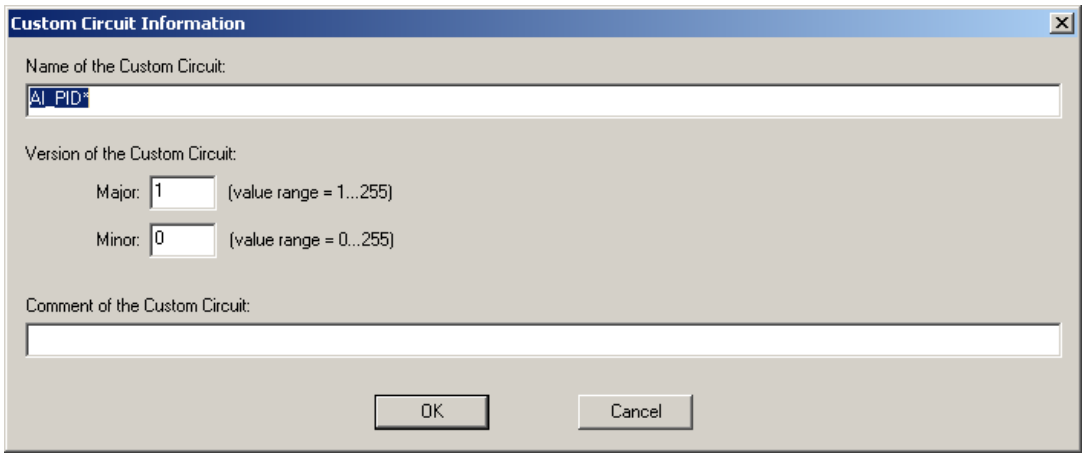
This command closes the CC definition window and returns to the solution program window.

If you have changed this CC definition, you will be asked if you want to preserve it.
Note: If you want to save this CC definition *permanently*, you must save the solution program in the solution program window.

CustomCircuit menu

Custom Circuit Information

This command displays a dialog box with the current name, version, and comment text of the current CC definition.

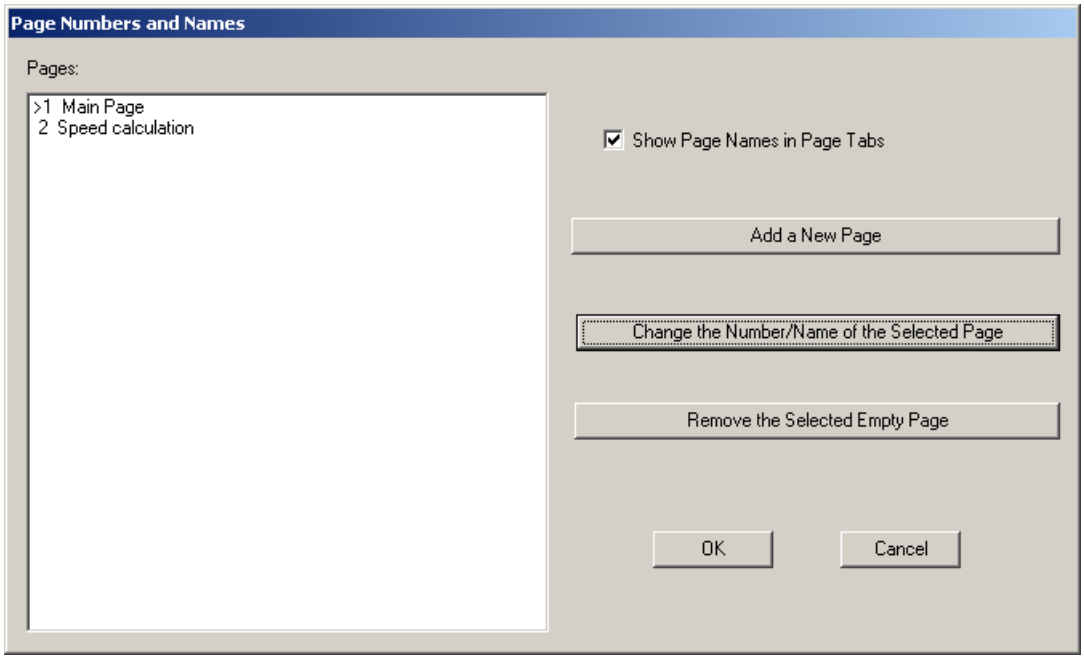


The CC name and version are displayed at the top of the CC definition window, too.

If needed, change these items and click **OK**.

Custom Circuit Pages

This command displays a dialog box with a list of the current CC definition pages.

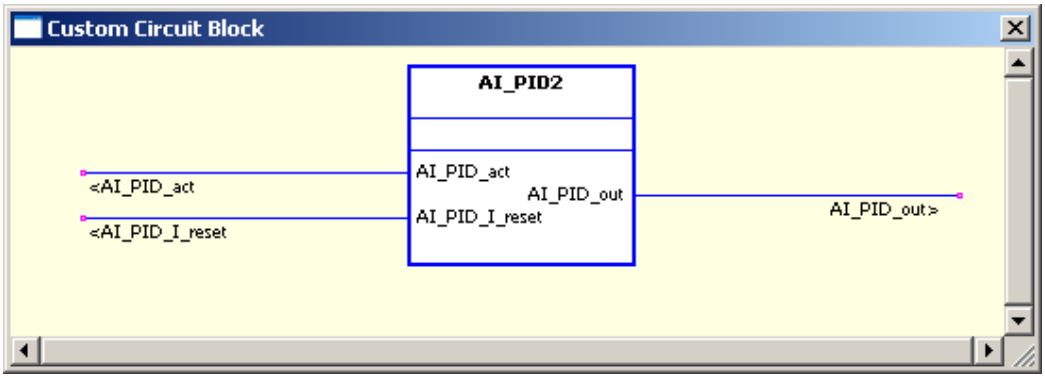


If needed, you can:

- Add new pages and specify numbers and names of new pages
- Change numbers and/or names of existing pages
- Remove empty pages
- Specify whether page names are displayed in page tabs. If page names are not displayed in the tabs, the page name of the tab can be seen as a *ToolTip* when the cursor is moved on the tab.

Custom Circuit Layout

This command (or the **F5** key) displays the *CC block window*. The instance block of the current CC definition is displayed in this window:



To find out the location of an external pin in the CC definition window, click on this pin in the CC block window.

Copy Page's Blocks

This command copies all blocks of the current page to DriveSPC's internal memory.

Paste Copied Blocks

This command pastes the blocks copied with command **Program > Copy Page's Blocks** or **CustomCircuit > Copy Page's Blocks** to an empty page.

You can copy/paste blocks:

- From a program page to another program page
- From a program page to a page of a custom circuit definition
- From a page of a custom circuit definition to a program page
- From a page of a custom circuit definition to another page of a custom circuit definition
- From a solution program to another solution program

Define Password for CC Display

If you want a password to be required to display this custom circuit, select this command and specify the password.

Help menu

User Interface

This command displays the main index of the *DriveSPC Help*.

If you press the **F1** key, help information about the currently active window, dialog box or menu item is displayed.

Block Information

This command opens the Firmware Manual of the drive in a separate PDF window.

If the cursor is on a normal block when you press the **F3** key, the description of this block is displayed in a separate PDF window.

About DriveSPC

This command displays the DriveSPC information box.

Menu commands (CC window, Structured Text)

Chapter overview

This chapter describes the menu commands of the custom circuit (Structured Text) definition window.

File menu

Print

This command prints the Structured Text program.

Return to the Main Program

This command closes the CC definition window and returns to the solution program window.

If you have changed this CC definition, you will be asked if you want to preserve it. *Note:* If you want to save this CC definition *permanently*, you must save the solution program in the solution program window.

Edit menu

Undo

This command undoes the last program change operation.

Cut

To cut text so you can move it to another location, select the text, and then select this menu command.

Copy

To copy text so you can paste it in another location, select the text, and then select this menu command.

Paste

To paste text you have cut or copied, place the cursor where you want to paste the text, and then select this menu command.

Delete

To delete text, select it, and then select this menu command.

Select All

This command selects the whole program text in the window.

Find

This command is used to find occurrences of a user-specified text string.

Replace

This command is used to replace one or more occurrences of a user-specified text string with another user-specified text string.

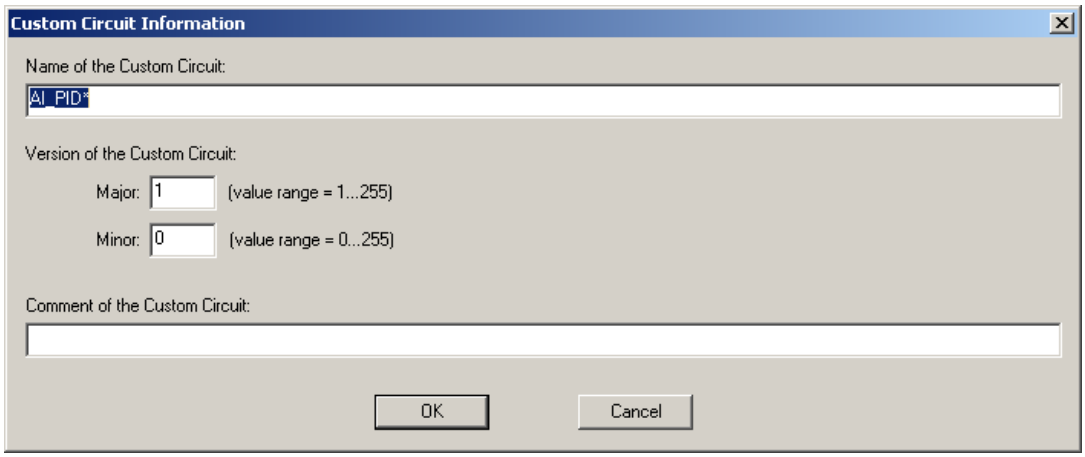
Go to Line

This command highlights the desired program line.

CustomCircuit menu

Custom Circuit Information

This command displays a dialog box with the current name, version, and comment text of the current CC definition.



The CC name and version are displayed at the top of the CC definition window, too.

If needed, change these items and click **OK**.

Check Program

This command checks the validity of the CC program.

If errors or warnings are found, they are listed in the Error window. You can go to the program line of an error by double-clicking on the desired error in the Error window.

Define Password for CC Display

If you want a password to be required displaying of this custom circuit, select this command and specify the password.

Help menu

User Interface

This command displays the main index of the *DriveSPC Help*.

If you press the **F1** key, help information about the currently active window, dialog box or menu item is displayed.

Structured Text Information

This command displays the main index of the Structured Text help.

If the cursor is on a Structured Text keyword when you press the **F3** key, the description of this keyword is displayed.

About DriveSPC

This command displays the DriveSPC information box.

Appendix 1

Appendix overview

This appendix summarises the usage of the **Ctrl**, **Alt**, and **Shift** keys with the mouse and the use of the **F3** key.

DriveSPC can be used without the **Ctrl**, **Alt**, and **Shift** keys. These keys are intended only to speed up the use of DriveSPC.

Usage of the Ctrl, Alt, and Shift keys with the left mouse button

Input pin

Ctrl Connect to the selected output pin (if any selected) or
select this pin for connection to an output pin (if no output pin selected)

Alt Go to the connected output pin (if any)

Shift Start/stop *On-Line* monitoring of this pin

Output pin

Ctrl Connect to the selected input pin (if any selected) or
select this pin for connection to an input pin (if no input pin selected)

Alt Display a list of the connected input pins (blocks).
If desired, you can now go to a connected input pin.

Shift Start/stop *On-Line* monitoring of this pin

Block instance

Ctrl *If a normal block or CC instance*:
Change the execution position of this block instance

Alt Select this block instance for block move

Empty screen location

Ctrl Add a new normal block instance here

Alt Put the selected block instance here (if any selected)

Wire

Alt Highlight this wire

Usage of the Ctrl and Shift keys with the right mouse button

Anywhere in the DriveSPC window

Ctrl Decrease the value of the zoom percentage

Shift Increase the value of the zoom percentage

Usage of the Ctrl and Shift keys with the mouse wheel

Anywhere in the DriveSPC window

Ctrl Increase/decrease the value of the zoom percentage

Shift Scroll horizontally

Usage of the F3 key

Pin of a firmware block

F3 Display the description of the pin parameter of this pin

Instance of a firmware/normal block

F3 Display the description of this block

Instance of a custom circuit definition

F3 Display the definition of this custom circuit

Instance of a parameter block

F3 Display the description of block's parameter

Instance of an alarm block

F3 Display the description of block's alarm

Instance of a fault block

F3 Display the description of block's fault

Structured Text keyword

F3 Display the description of this keyword

Appendix 2

Appendix overview

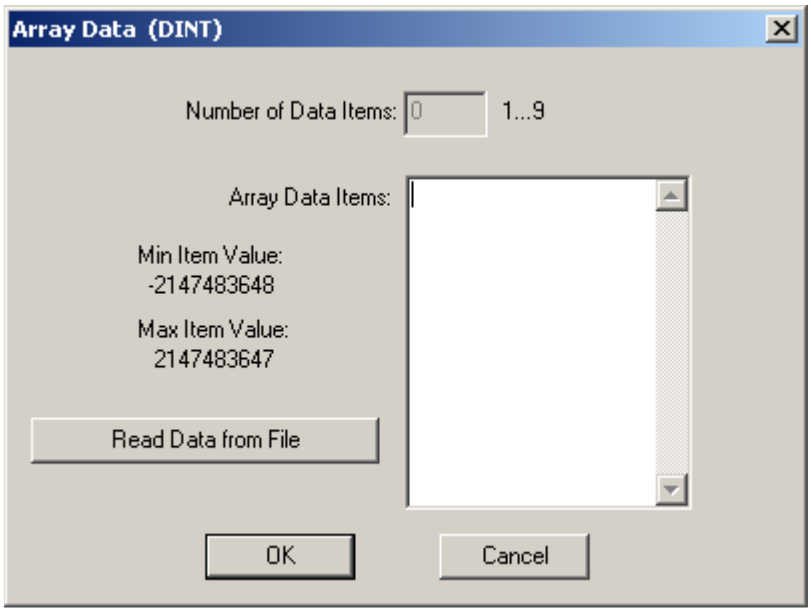
This appendix describes the definition of the data items of the array/structure that is in an input or output pin of a normal block.

Array

An array is used when the data type of all data items is the same.

To define pin array data:

1. Right-click on the desired input or output pin of a normal block or CC instance, and select **Define Pin Array Data**.
   A dialog box is displayed with a list of current array items (if any).



2. Modify the items in the list by entering the data items manually to the list or read the data items from a text file by clicking **Read Data from File**.
   The filename extension of the file must be `ARR`.
   Every data item in this file is on its own line, for example:

```
123.45
-7.1
45.9
```

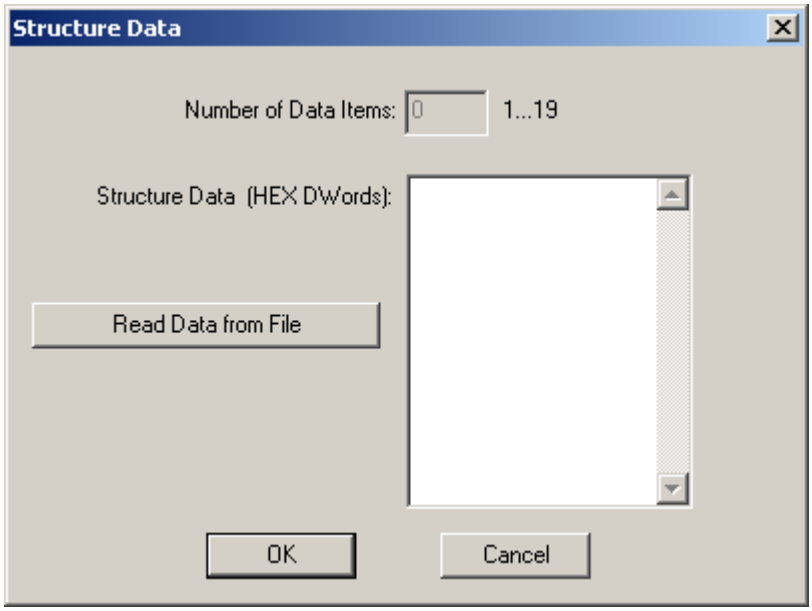
3. After you have made the modifications, click **OK**.

Structure

A structure is used when the data types of data items are not the same.

To define pin structure data:

1. Right-click on the desired input or output pin of a normal block or CC instance, and select **Define Pin Structure Data**.
   A dialog box is displayed with a list of current structure items (if any). All structure items are always 32-bit hexadecimal values.



2. Modify the items in the list by entering the data items manually to the list (32-bit hex values) or read the data items from a text file by clicking **Read Data from File**.
   The filename extension of the file must be `STR`.
   Every 32-bit hex data item in this file is on its own line, for example:

```
12345
ABCDEF12
76FF4
987A
```

3. After you have made the modifications, click **OK**.

Appendix 3

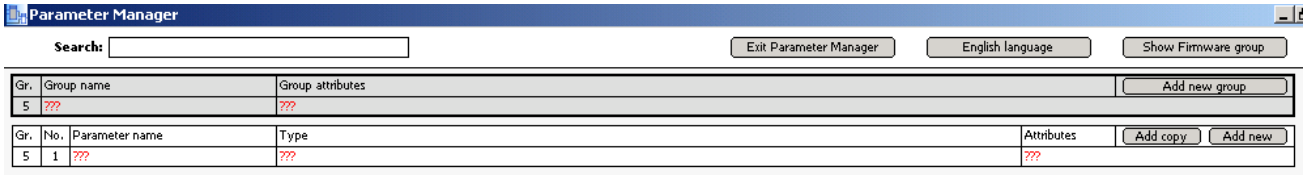
Appendix overview

This appendix describes the usage of the Parameter Manager window.

Parameter Manager

The Parameter Manager window is opened by the menu command **Program > Parameter Manager**.

If there are no user parameters defined when the Parameter Manager window is opened, there is one (undefined) group and parameter in the window as follows:



The **Search** field at the top is used for parameter or group search. You can type a partial parameter or group name, or a group and index number, and the parameter list is moved to the location of the first found parameter or group.
If the **Next** and **Prev** buttons (after the **Search** field) are visible, they can be used to search next and previous hits.

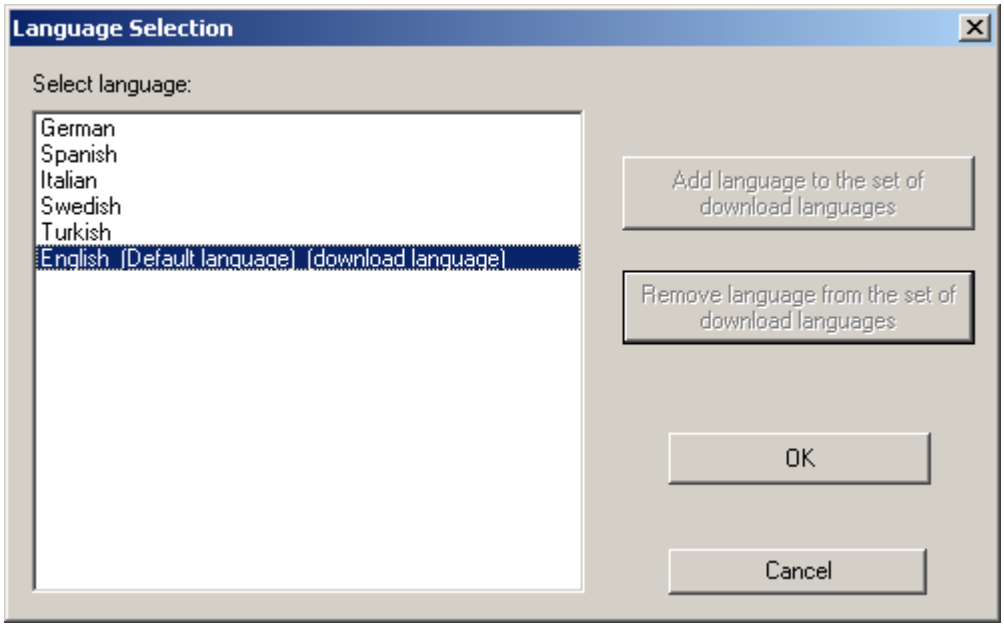
Languages of group and parameter texts

The **Language** button at the top of the window is used for two purposes:

- Selection of the languages that belong to the set of download languages.
  When this solution program is downloaded to a drive, the group and parameter texts of the download languages are included in the download.
  After the download, the desired language of these downloaded languages can be selected in the drive by the language selection parameter.
- Selection of the currently active language. The selected language must belong to the set of download languages.
  Texts of the selected language are now displayed in the textual items of the Parameter Manger window. If a textual item has no text defined in the selected language, its text in the default language (if such text exists) is displayed.
  All texts that are now written to the items of the Parameter Manager will belong to the selected language.

The name of the currently active language is displayed in the **Language** button.

Click the **Language** button. A dialog box is displayed:



Selection of the download languages

English is the default language, and it is always one of the download languages.

Other languages can be added to the set of download languages by the **Add language to the set of download languages** button.

Languages can be removed from the set of download languages by the **Remove language from the set of download languages** button.
The default language cannot be removed.

Finally, select one of the download languages and click **OK**.
The selected language will be the currently active language. See below.

Selection of the currently active language

Select the desired language from the set of download languages. Click **OK**.

Text in the selected language is now displayed in the textual items of the Parameter Manager window.
If a textual item has no text defined in the selected language, its text in the default language (if such text exists) is displayed.

Any text that is now entered to the items of the Parameter Manager will belong to the selected language.

Groups and parameters

Every group and parameter contains two lines:

- Header line
- Value line

Groups are added to Parameter Manager by the following two buttons:

- A new group is created by the **Add new group** button in the header line of the group after which the new group will be added.
- An existing firmware group is displayed for parameter addition by the **Show Firmware group** button at the top of the window.

New parameters are added to a group by the two buttons in the header line of the parameter after which the new parameter will be added:

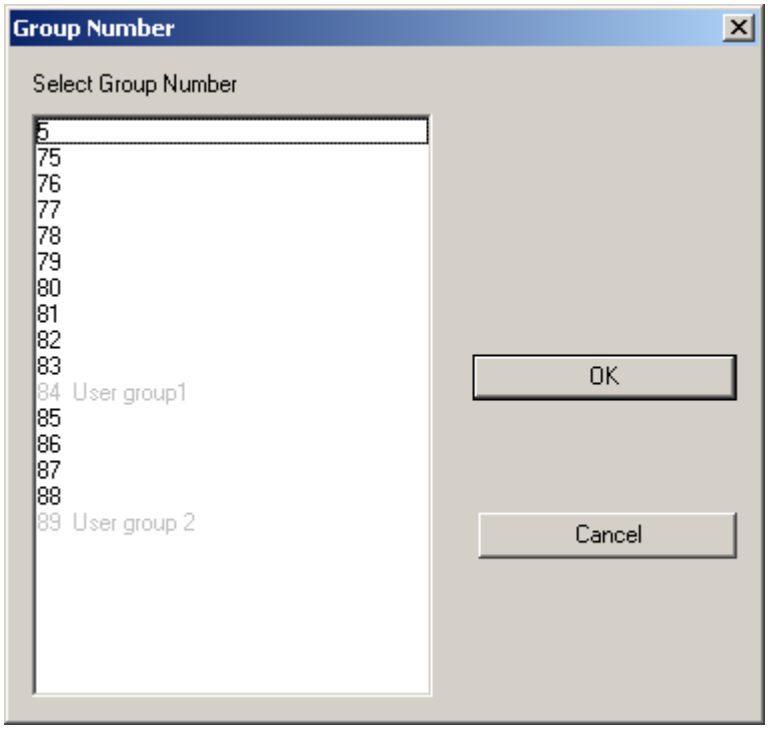
- A completely new parameter is added by the **Add new** button.
- A copy of the clicked parameter is added by the **Add copy** button.

The parameter or group is copied, pasted, or deleted by right-clicking its **header line**.
Value items in a value line are set/changed by left-clicking the desired **value area**.

Value items of groups

Group number

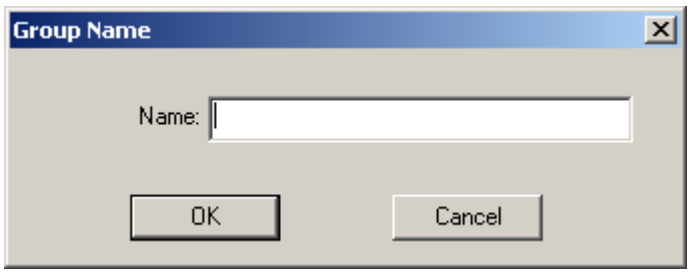
Group number is changed by left-clicking the group's **Gr** field. A dialog box is displayed:



Select the desired group number and click **OK**.

Group name

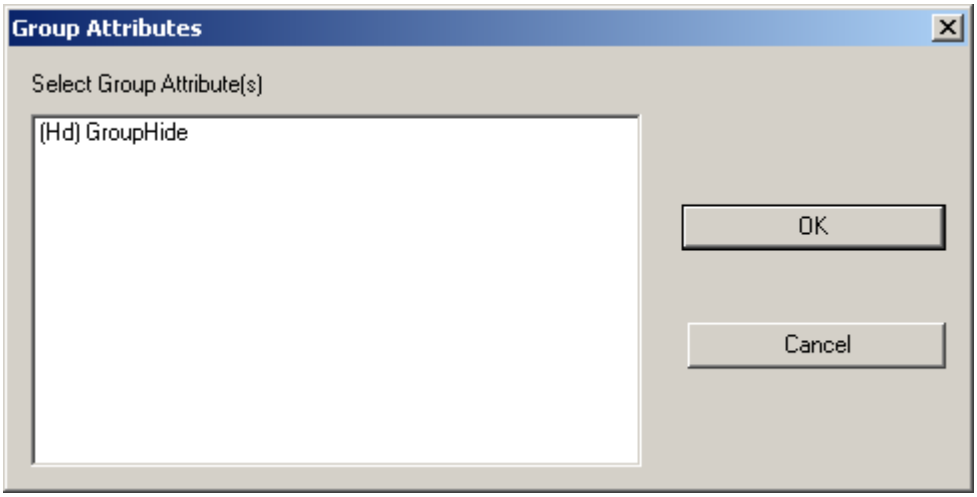
Group name is specified by left-clicking the group's **Group name** field. A dialog box is displayed:



Enter the desired group name and click **OK**.

Group attributes

Group attributes are set by left-clicking the group's **Group attributes** field. A dialog box is displayed:

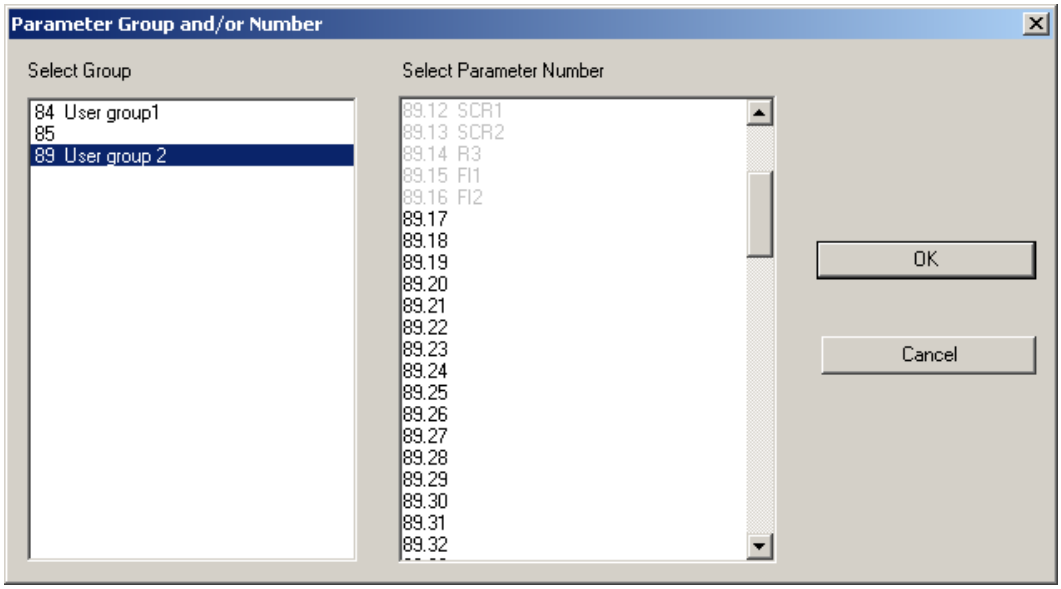


Select the desired group attributes (if any). Click **OK**.

Common value items of parameters

Group number

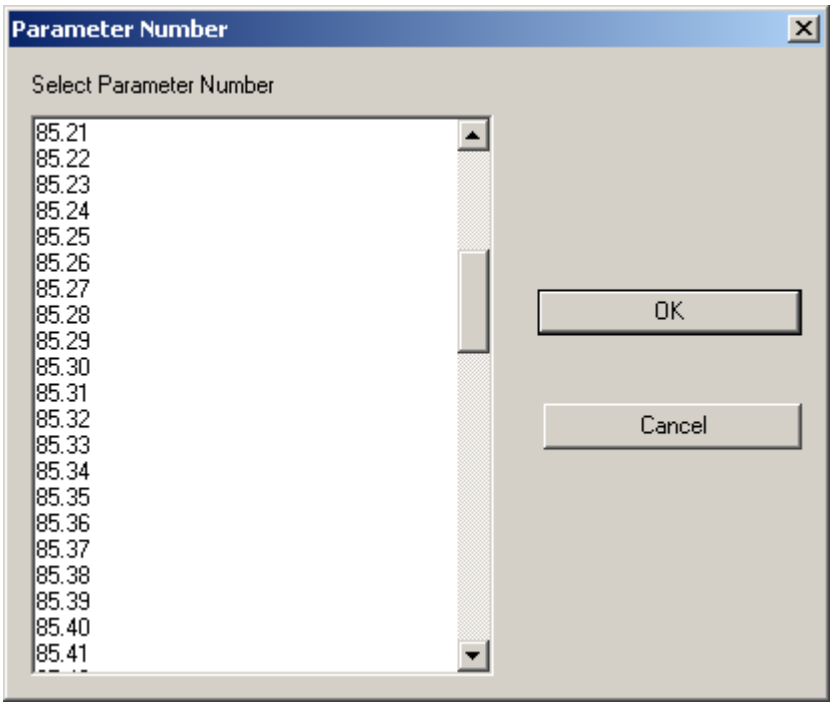
Group number of a parameter is changed by left-clicking the parameter's **Gr** field. A dialog box is displayed:



Select the desired group number and the desired parameter number in the selected group. Click **OK**.

Parameter number

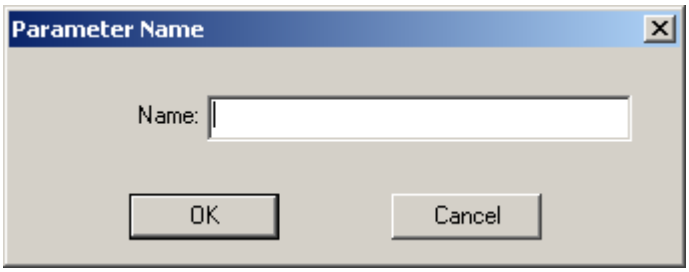
Parameter number (index) is changed by left-clicking the parameter's **No.** field. A dialog box is displayed:



Select the desired parameter number and click **OK**.

Parameter name

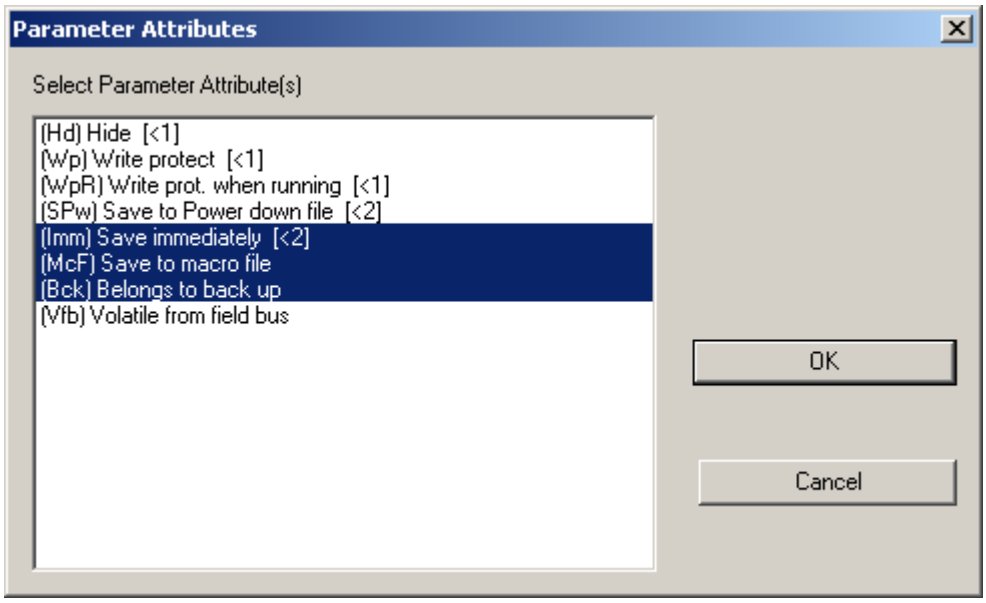
Parameter name is specified by left-clicking the parameter's **Parameter name** field. A dialog box is displayed:



Enter the desired parameter name. Click **OK**.

Parameter attributes

Parameter attributes are set by left-clicking the parameter's **Attributes** field. A dialog box is displayed:



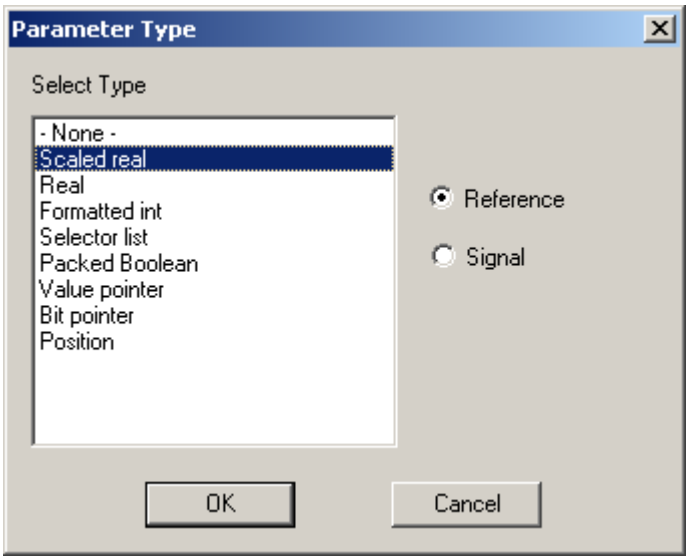
If there is string `[<n]` (e.g. `[<1]`) after an attribute's name, it means that no more than one of the attributes marked with the same string (e.g., `[<1]`) can be selected.

If there is string `[>n]` (e.g. `[>1]`) after an attribute's name, it means that at least one of the attributes marked with the same string (e.g. `[>1]`) must be selected.

Select the desired parameter attributes (if any). Click **OK**.

Parameter type

Parameter type is specified by left-clicking the parameter's **Type** field. A dialog box is displayed:



Some parameter types allow the *Reference/Signal* selection.
Reference parameter is a parameter that can be written by users.
Signal parameter is a parameter that can only be read by users.

Select the desired parameter type (and *Reference/Signal* if applicable). Click **OK**.

New type-specific value items (see below) are added to the parameter line after the type selection.

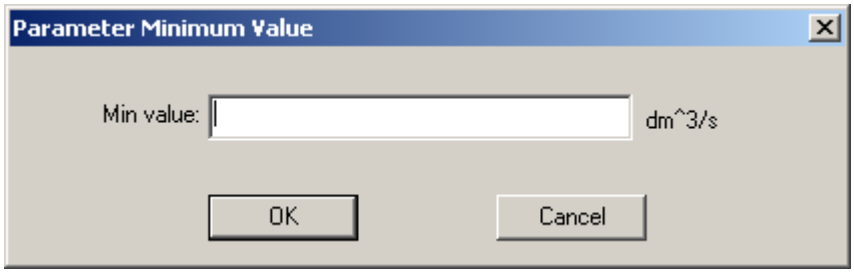
Type-specific value items of Scaled Real parameters

Gr.	No.	Parameter name	Type	Min	Max	Default	Scale internal	Scale panel	Format	Attributes		
89	17	User parameter 1	Scaled real ref	???	???	???	???	???	???	???		

Gr.	No.	Parameter name	Type	Scale internal	Scale panel	Format	Attributes		
89	18	User parameter 2	Scaled real sig	???	???	???	???		

Minimum value (Reference parameter only)

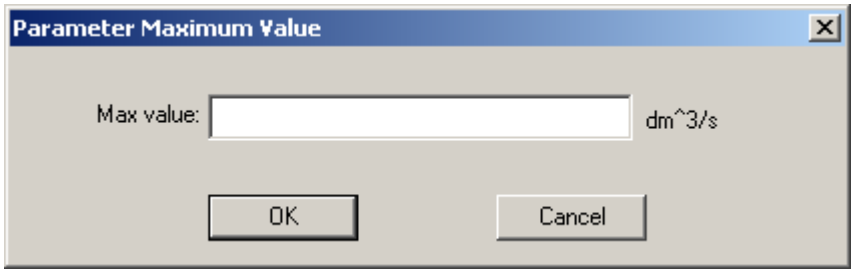
Minimum value is specified by left-clicking the parameter's **Min** field. A dialog box is displayed:



Enter the desired value. Click **OK**.

Maximum value (Reference parameter only)

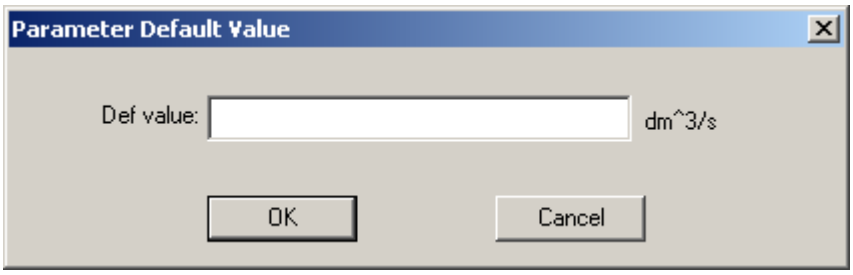
Maximum value is specified by left-clicking the parameter's **Max** field. A dialog box is displayed:



Enter the desired value. Click **OK**.

Default value (Reference parameter only)

Default value is specified by left-clicking the parameter's **Default** field. A dialog box is displayed:

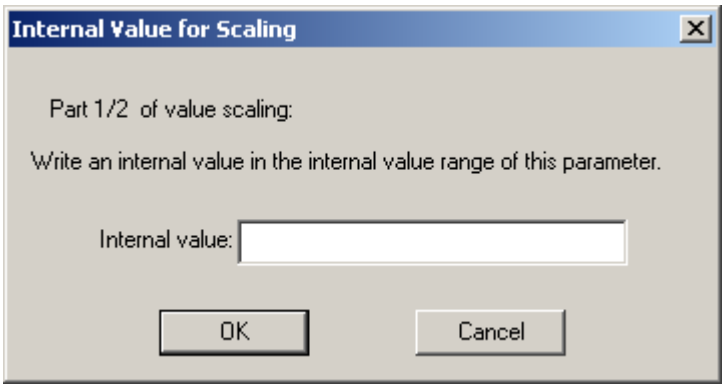


Enter the desired value. Click **OK**.

Value scaling part 1/2: Scale internal

You have to specify how the parameter's internal values are displayed to users. This is called value scaling. It is a two-step procedure.

The first step is started by left-clicking the parameter's **Scale internal** field. A dialog box is displayed:

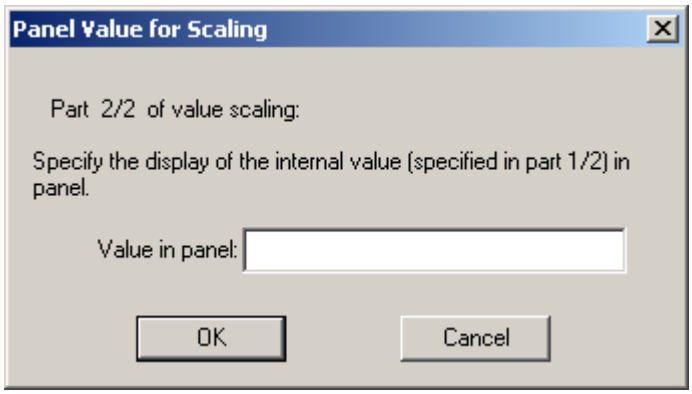


Enter the desired value (e.g. 1234.5). Click **OK**.

Value scaling part 2/2: Scale panel

You have to specify how the parameter's internal values are displayed to users. This is called value scaling. It is a two-step procedure.

The second step is started by left-clicking the parameter's **Scale panel** field. A dialog box is displayed:

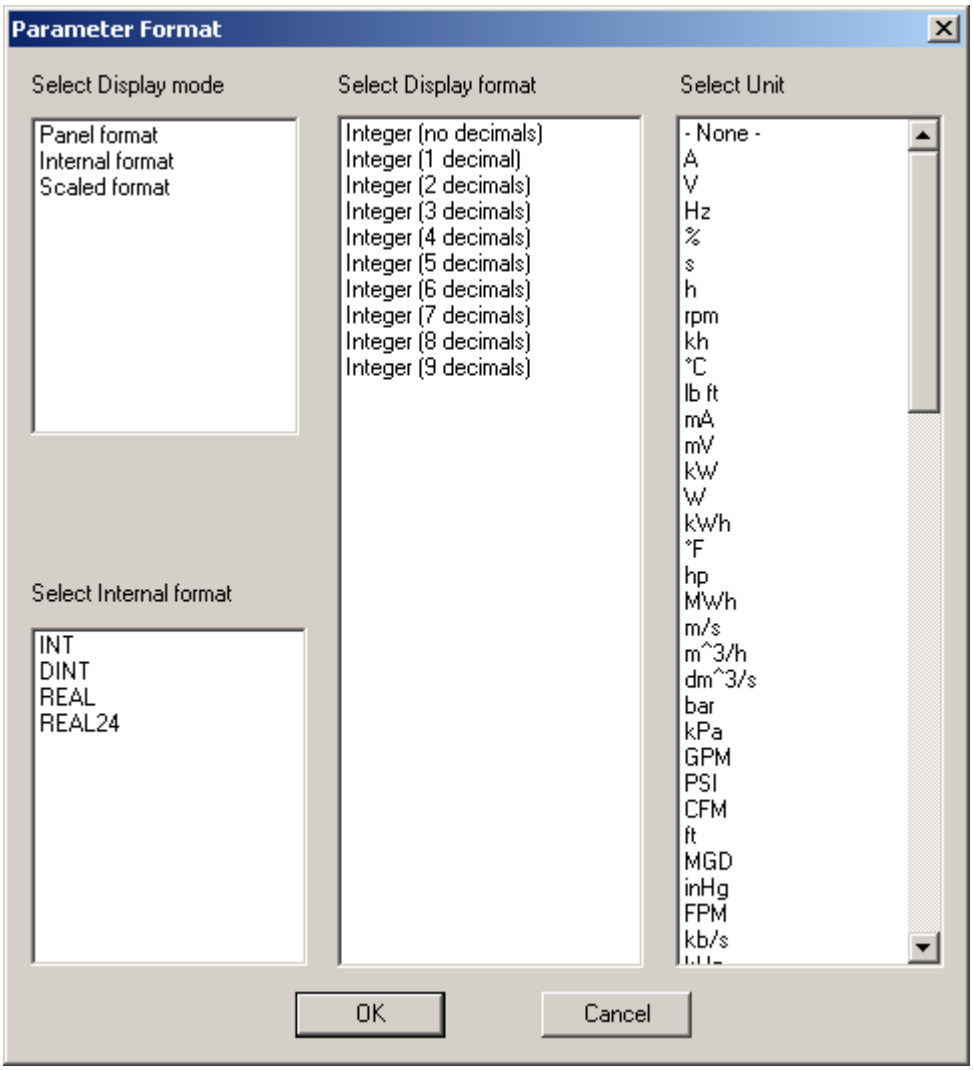


Enter the desired value (e.g. 100). Click **OK**.

Example: If you wrote value 1234.5 in step one and value 100 in step two, it means that the parameter's internal value of 1234.5 is displayed to users as value 100. All other internal values of this parameter are scaled in the same way.

Value format

Value format is specified by left-clicking the parameter's **Format** field. A dialog box is displayed:



The meaning of the Display mode alternatives is as follows:

- *Panel format* is a value format that includes scaling, selected Display format, and selected Unit (this format is used, e.g., by panel).
- *Scaled format* is a value format that includes only scaling.
- *Internal format* is the value format used by the internal program code.

Minimum, maximum, and default values in the value line of this parameter are displayed according to the selected Display mode.

Select the desired items from the lists and click **OK**.

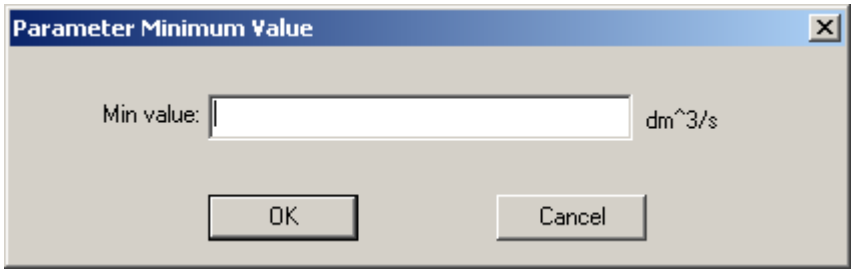
Type-specific value items of Real parameters

Gr.	No.	Parameter name	Type	Min	Max	Default	Format	Attributes		
89	20	User parameter 4	Real ref	???	???	???	???	???		

Gr.	No.	Parameter name	Type	Format	Attributes		
89	21	User parameter 5	Real sig	???	???		

Minimum value (Reference parameter only)

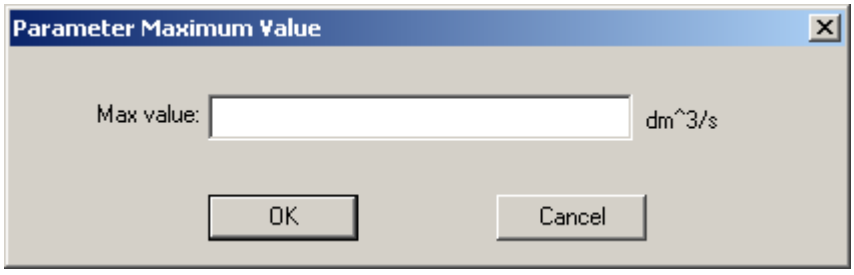
Minimum value is specified by left-clicking the parameter's **Min** field. A dialog box is displayed:



Enter the desired value. Click **OK**.

Maximum value (Reference parameter only)

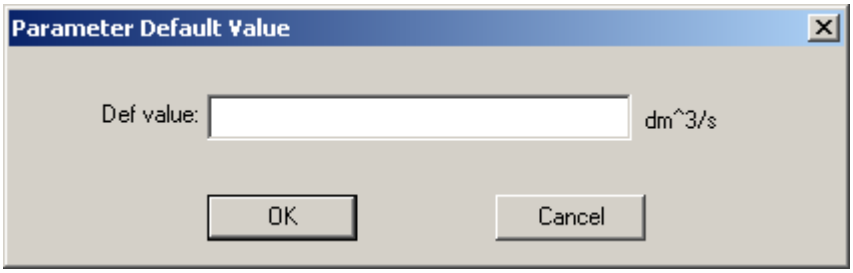
Maximum value is specified by left-clicking the parameter's **Max** field. A dialog box is displayed:



Enter the desired value. Click **OK**.

Default value (Reference parameter only)

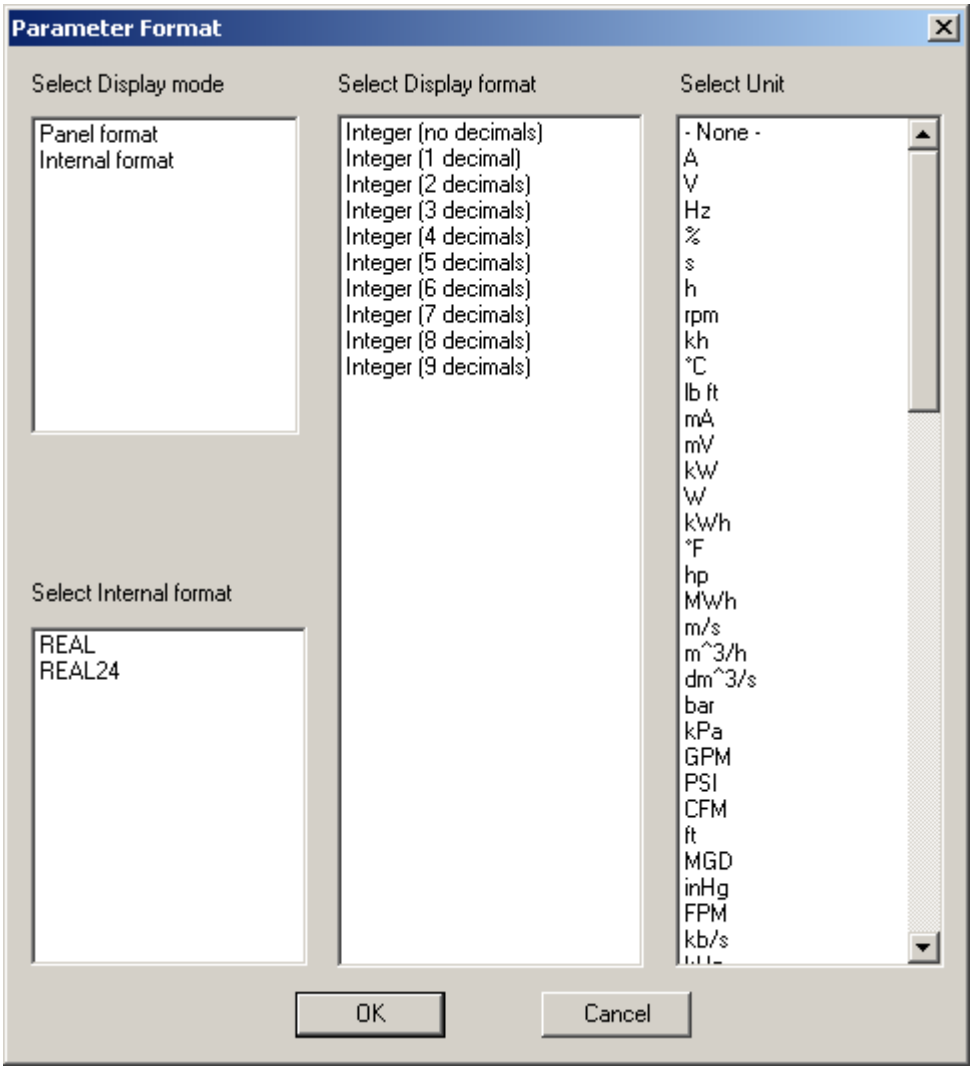
Default value is specified by left-clicking the parameter's **Default** field. A dialog box is displayed:



Enter the desired value. Click **OK**.

Value format

Value format is specified by left-clicking the parameter's **Format** field. A dialog box is displayed:



The meaning of the Display mode alternatives is as follows:

- *Panel format* is a value format that includes the selected Display format and selected Unit (this format is used, e.g., by panel).
- *Internal format* is the value format used by the internal program code.

Minimum, maximum, and default values in the value line of this parameter are displayed according to the selected Display mode.

Select the desired items from the lists. Click **OK**.

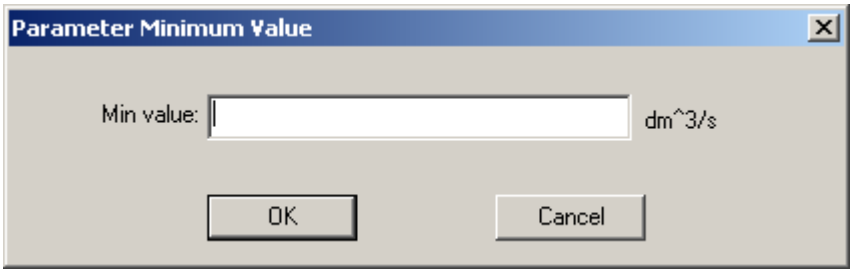
Type-specific value items of Formatted integer parameters

Gr.	No.	Parameter name	Type	Min	Max	Default	Format	Attributes		
89	22	User parameter 6	Formatted int ref	???	???	???	???	???		

Gr.	No.	Parameter name	Type	Format	Attributes		
89	23	User parameter 7	Formatted int sig	???	???		

Minimum value (Reference parameter only)

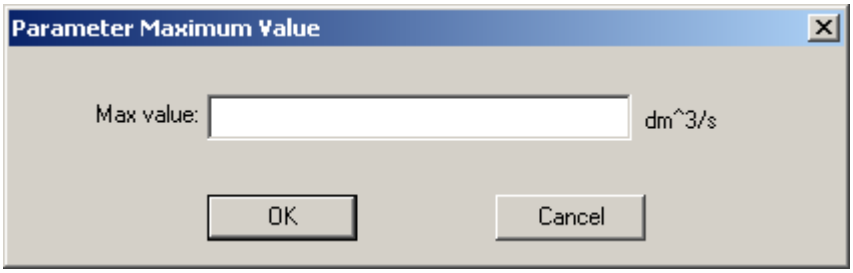
Minimum value is specified by left-clicking the parameter's **Min** field. A dialog box is displayed:



Enter the desired value. Click **OK**.

Maximum value (Reference parameter only)

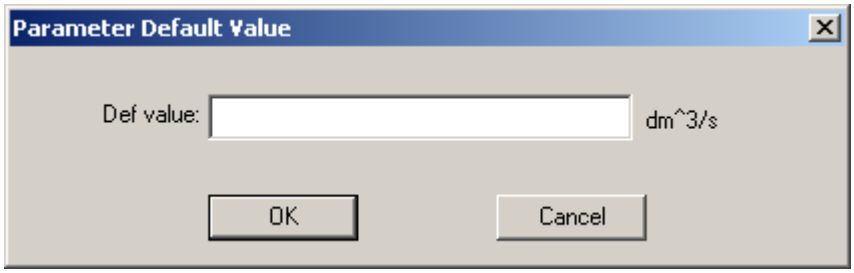
Maximum value is specified by left-clicking the parameter's **Max** field. A dialog box is displayed:



Enter the desired value. Click **OK**.

Default value (Reference parameter only)

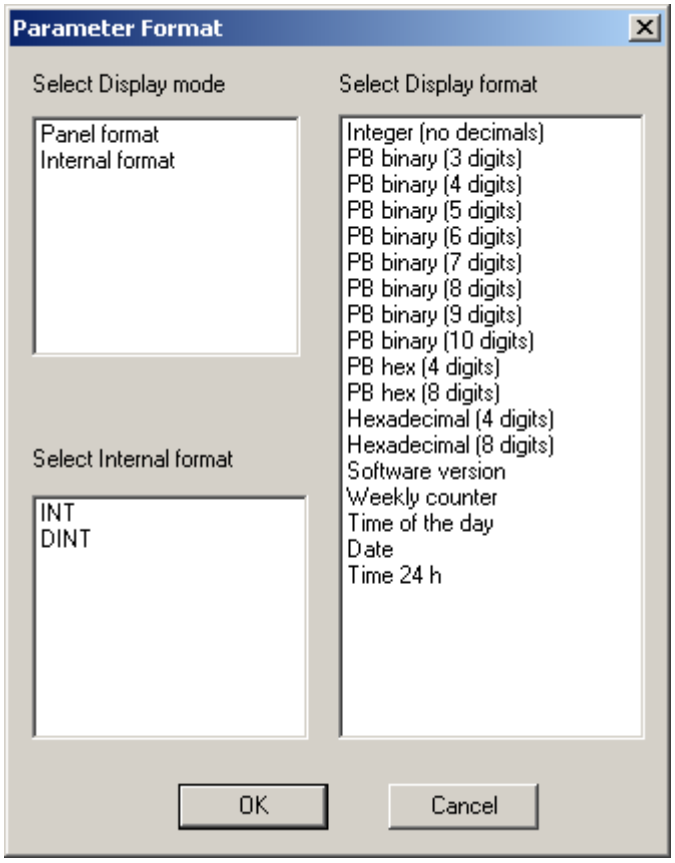
Default value is specified by left-clicking the parameter's **Default** field. A dialog box is displayed:



Enter the desired value. Click **OK**.

Value format

Value format is specified by left-clicking the parameter's **Format** field. A dialog box is displayed:



The meaning of the Display mode alternatives is as follows:

- *Panel format* is a value format that includes the selected Display format (this format is used, e.g., by panel).
- *Internal format* is the value format used by the internal program code.

Minimum, maximum, and default values in the value line of this parameter are displayed according to the selected Display mode.

Select the desired items from the lists. Click **OK**.

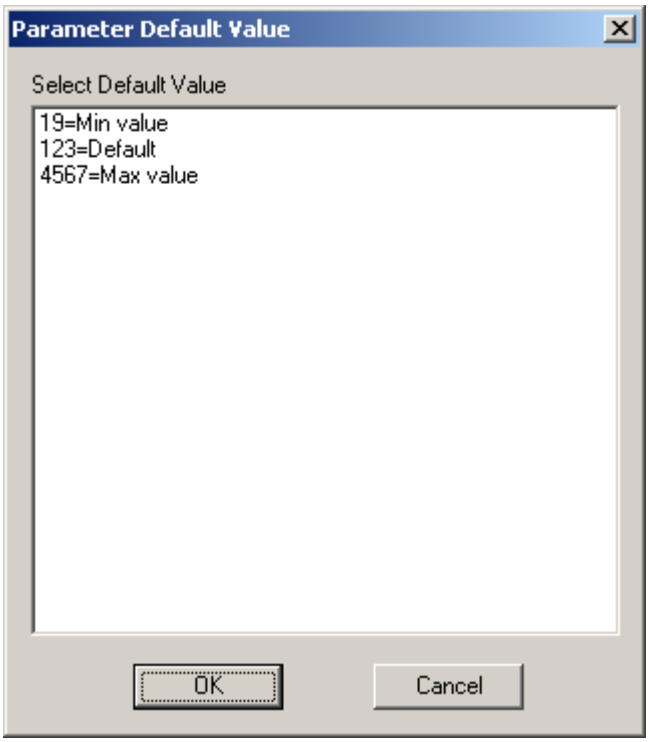
Type-specific value items of Selector list parameters

Gr.	No.	Parameter name	Type	Default	Selection items	Attributes		
89	24	User parameter 8	Selector list ref	???	???	???		

Gr.	No.	Parameter name	Type	Selection items	Attributes		
89	25	User parameter 9	Selector list sig	???	???		

Default value (Reference parameter only)

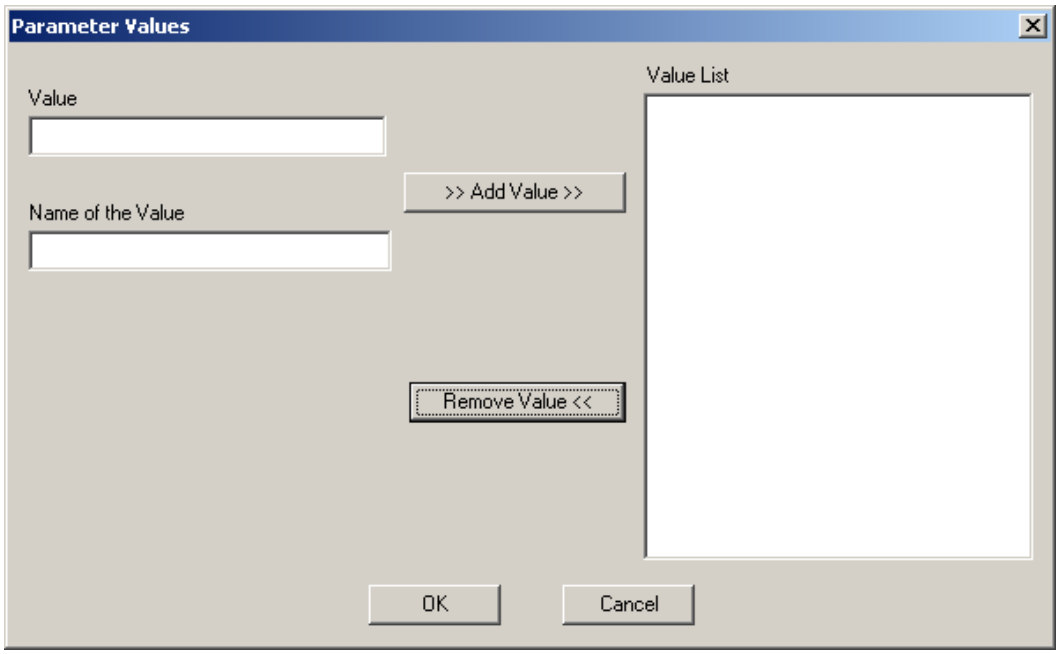
Default value is specified by left-clicking the parameter's **Default** field. A dialog box is displayed with the list of user-specified selectable values:



Select the desired value. Click **OK**.

Selection items

Selectable values are specified by left-clicking the parameter's **Selection items** field. A dialog box is displayed:



In order to add a selectable value to the Value list, enter the value in the **Value** field, the value name in the **Name of the Value** field and click the **Add Value** button.

If you want to change the name of an existing item in the Value list, click on the desired item in the list, change the name in the **Name of the Value** field and click the **Add Value** button.
Use of multiple languages: If you have already written value names in some language and you have selected another current language, you can write new value names in the selected language in this way.
The new names replace the previous names on the screen but the previous names can be seen again by selecting the previous language as the current language.

In order to remove a value from the Value list, select the item to be removed from the list and click the **Remove Value** button.

When you have completed the Value list, click **OK**.

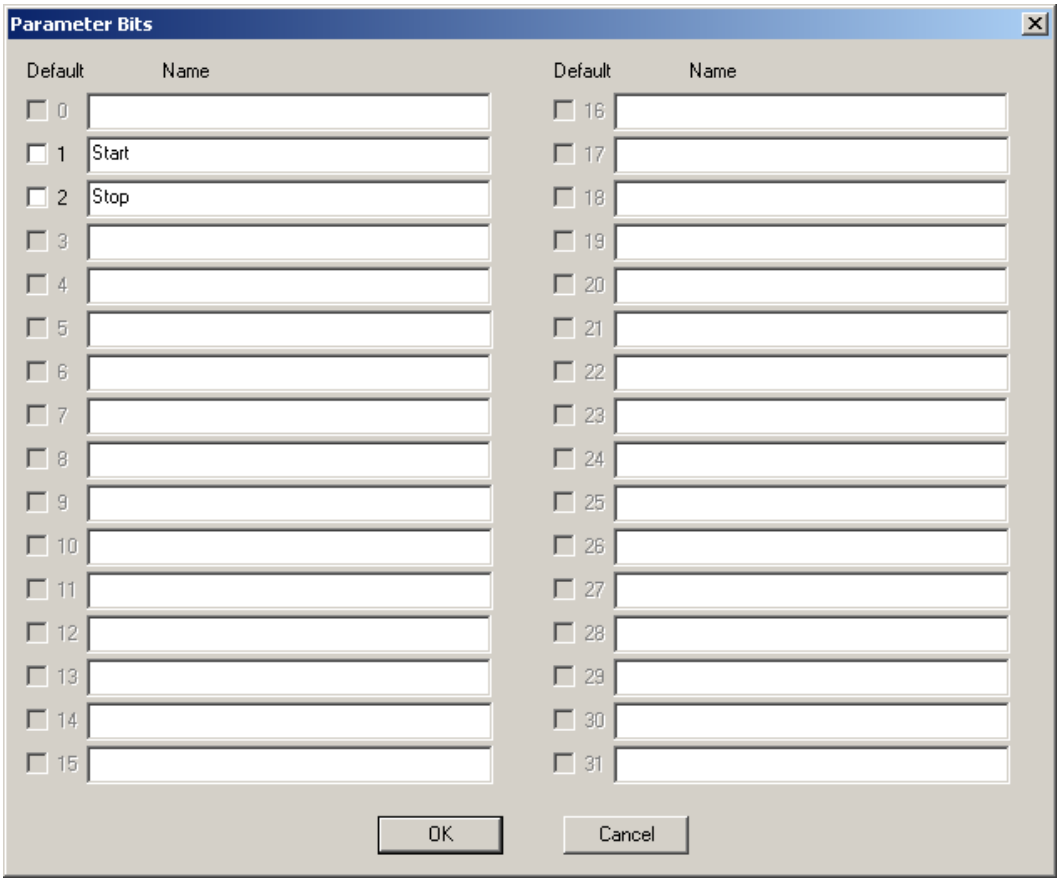
Type-specific value items of Packed Boolean parameters

Gr.	No.	Parameter name	Type	Bit list	Attributes		
89	26	User parameter 10	Packed Boolean ref	???	???		

Gr.	No.	Parameter name	Type	Bit list	Attributes		
89	27	User parameter 11	Packed Boolean sig	???	???		

Bit list (Reference parameter)

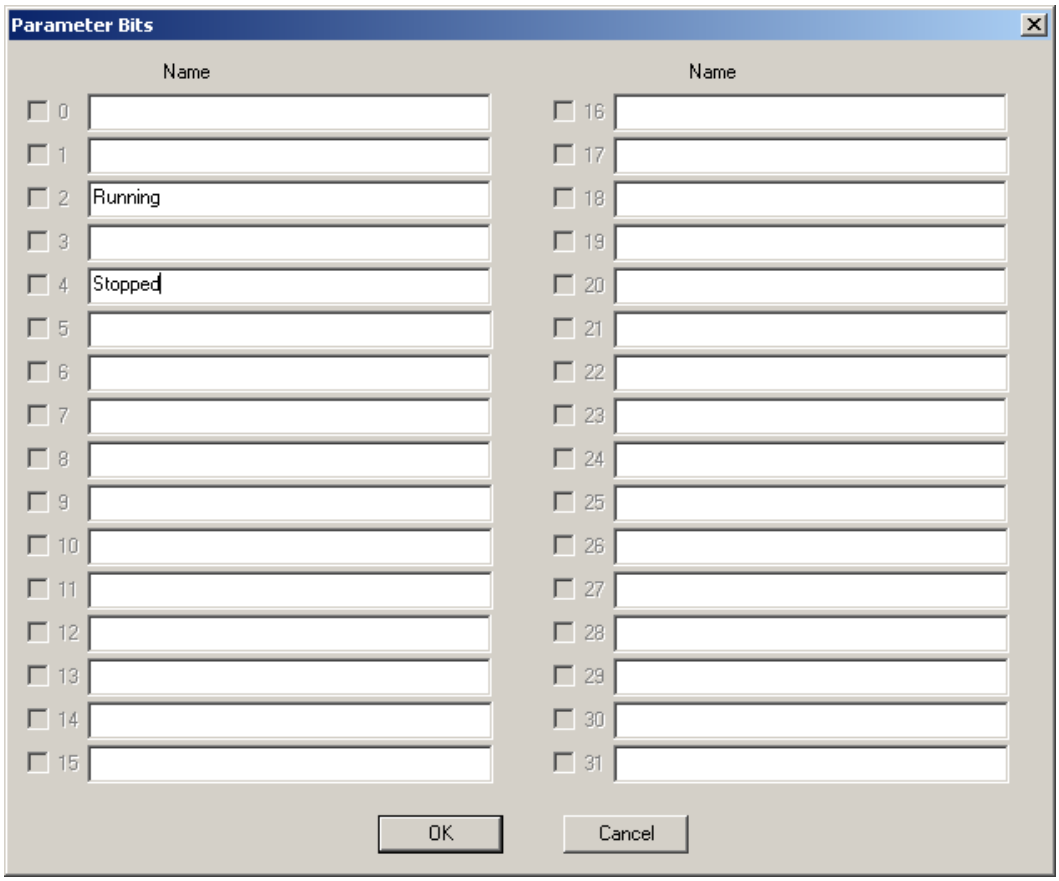
Bit names and default bits are specified by left-clicking the parameter's **Bit list** field. A dialog box is displayed:



Enter the desired bit names and select the desired default bits. Click **OK**.

Bit list (Signal parameter)

Bit names are specified by left-clicking the parameter's **Bit list** field. A dialog box is displayed:



Enter the desired bit names. Click **OK**.

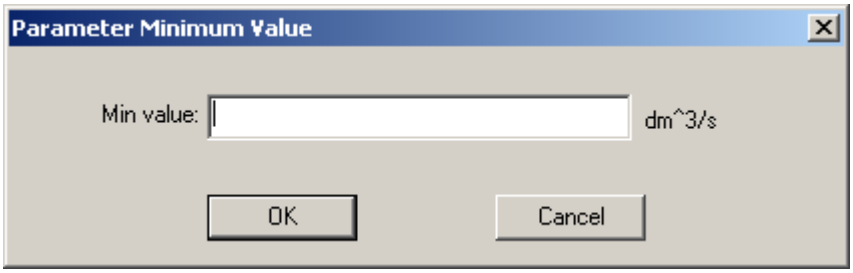
Type-specific value items of Position reference parameters

Gr.	No.	Parameter name	Type	Min	Max	Default	Attributes		
89	28	User parameter 12	Position ref	???	???	???	???		

Gr.	No.	Parameter name	Type	Attributes		
89	29	User parameter 13	Position sig	???		

Minimum value

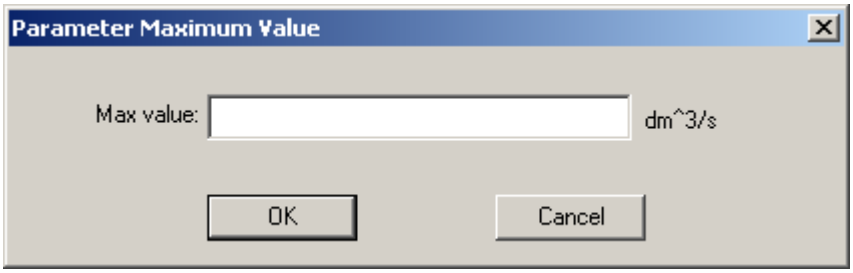
Minimum value is specified by left-clicking the parameter's **Min** field. A dialog box is displayed:



Enter the desired value. Click **OK**.

Maximum value

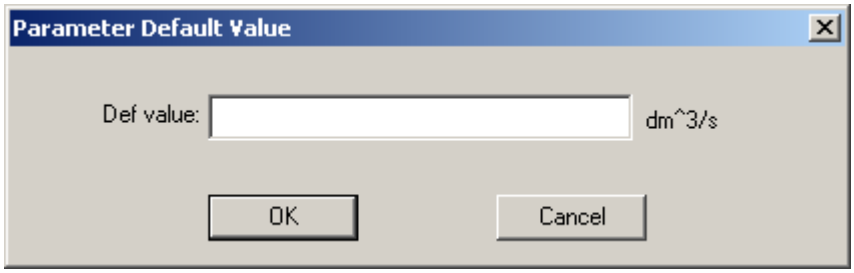
Maximum value is specified by left-clicking the parameter's **Max** field. A dialog box is displayed:



Enter the desired value. Click **OK**.

Default value

Default value is specified by left-clicking the parameter's **Default** field. A dialog box is displayed:



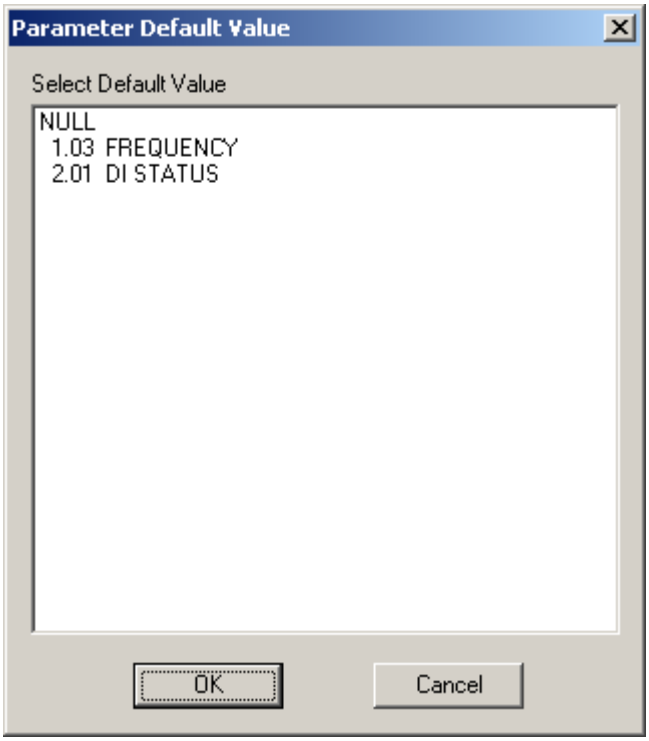
Enter the desired value. Click **OK**.

Type-specific value items of Value pointer parameters

Gr.	No.	Parameter name	Type	Default	Pointer list	Attributes		
89	30	User parameter 14	Value pointer	???	???	???		

Default value

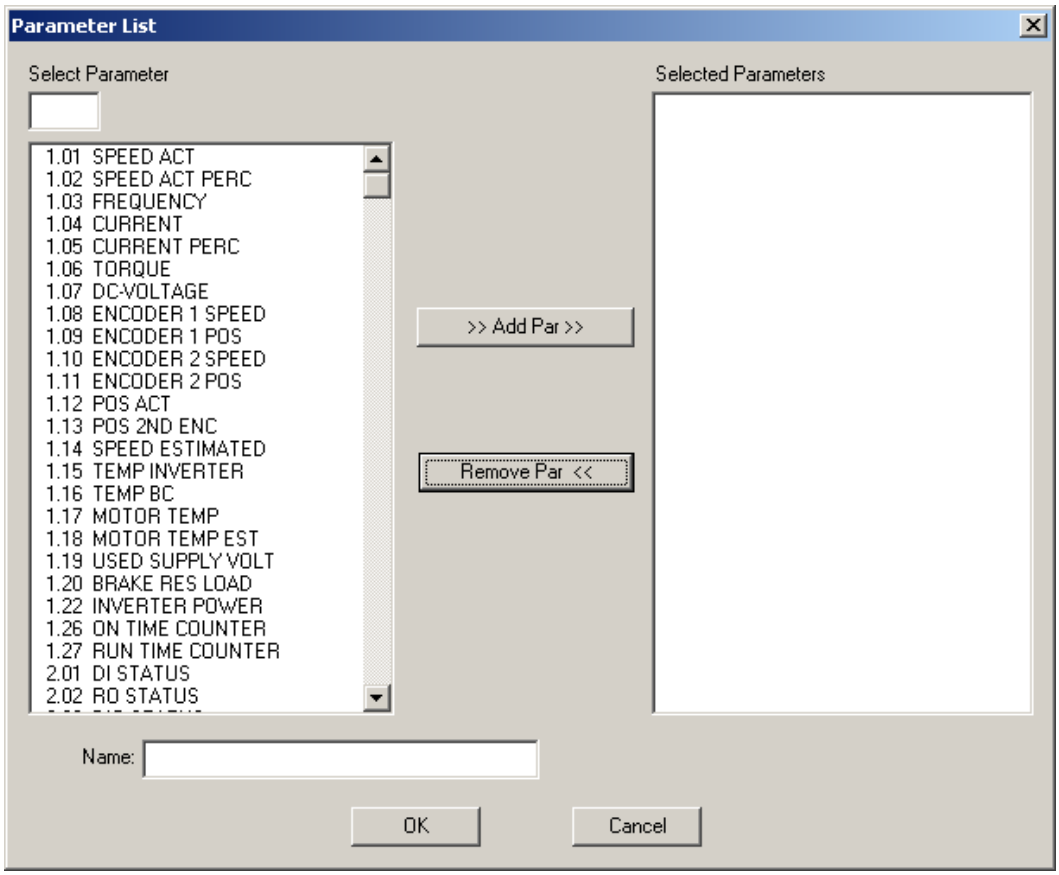
Default value is specified by left-clicking the parameter's **Default** field. A dialog box is displayed with the list of user-specified selectable values:



Select the desired value (value NULL means that no default value is required). Click **OK**.

Pointer list

Selectable pointer values are specified by left-clicking the parameter's **Pointer list** field. A dialog box is displayed:



In order to add a selectable pointer value to the Selected Parameters list, select a parameter from the parameter list, enter the desired name of this parameter to the **Name** field, and click the **Add Par** button.

If you want to change the name of an existing item in the Selected Parameters list, click on the desired item in the list, change the name in the **Name** field, and click the **Add Par** button.
Use of multiple languages: If you have already written names in some language and you have selected another current language, you can write new names in the selected language in this way.
The new names replace the previous names on the screen but the previous names can be seen again by selecting the previous language as the current language.

In order to remove a parameter from the Selected Parameters list, select the item to be removed from the list and click the **Remove Par** button.

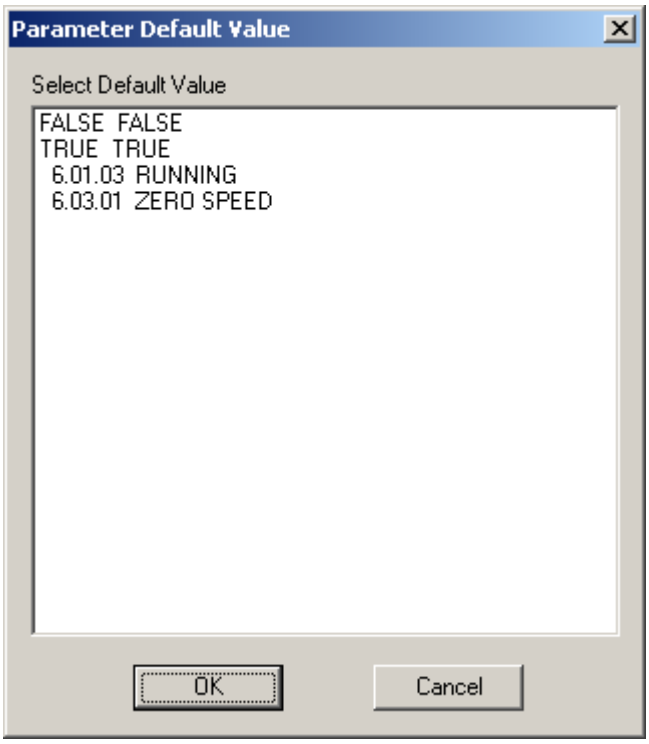
When you have completed the Selected Parameters list, click **OK**.

Type-specific value items of Bit pointer parameters

Gr.	No.	Parameter name	Type	Default	Pointer list	Attributes	Add copy	Add new
89	31	User parameter 15	Bit pointer	???	???	???		

Default value

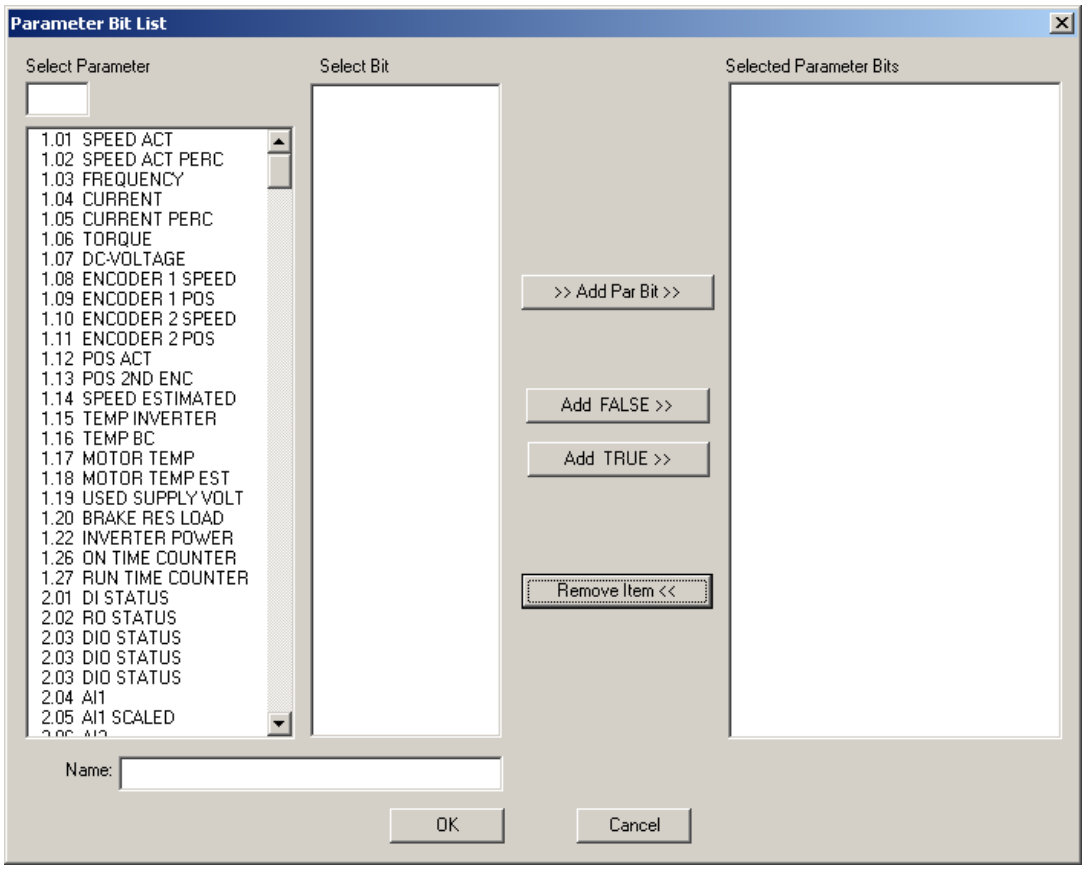
Default value is specified by left-clicking the parameter's **Default** field. A dialog box is displayed with the list of user-specified selectable values:



Select the desired value. Click **OK**.

Pointer list

Selectable pointer values are specified by left-clicking the parameter's **Pointer list** field. A dialog box is displayed:



In order to add a selectable bit value to the Selected Parameter Bits list, select a parameter from the parameter list, a bit from the bit list, enter the desired name of this parameter bit to the **Name** field, and click the **Add Par Bit** button.

In order to add value FALSE to the Selected Parameter Bits list, enter the desired name of the FALSE value to the **Name** field, and click the **Add FALSE** button.

In order to add value TRUE to the Selected Parameter Bits list, enter the desired name of the TRUE value to the **Name** field, and click the **Add TRUE** button.

If you want to change the name of an existing item in the Selected Parameter Bits list, click on the desired item in the list, change the name in the **Name** field, and click the **Add Par Bit** or **Add FALSE** or **Add TRUE** button.
Use of multiple languages: If you have already written names in some language and you have selected another current language, you can write new names in the selected language in this way.
The new names replace the previous names on the screen but the previous names can be seen again by selecting the previous language as the current language.

In order to remove an item from the Selected Parameters list, select the item to be removed from the list and click the **Remove Item** button.

When you have completed the Selected Parameter Bits list, click **OK**.

Appendix 4

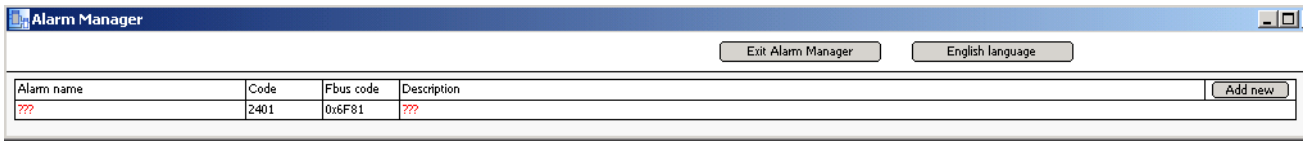
Appendix overview

This appendix describes the usage of the Alarm Manager window.

Alarm Manager

The Alarm Manager window is opened by the menu command **Program > Alarm Manager**.

If there are no user alarms defined when the Alarm Manager window is opened, there is one (undefined) alarm in the window as follows:



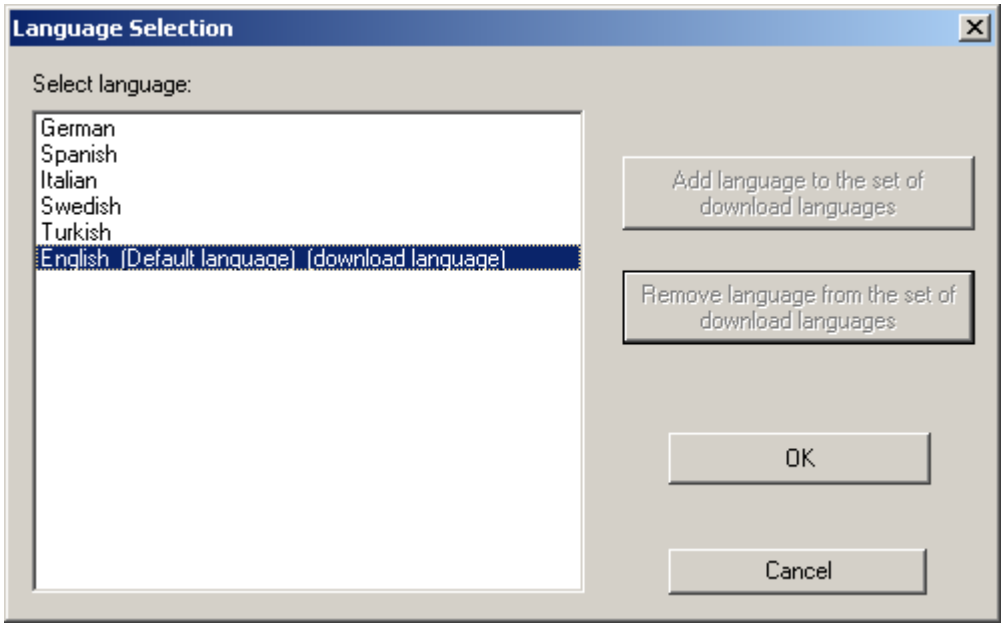
Languages of alarm texts

The **Language** button at the top of the window is used for two purposes:

- Selection of the languages that belong to the set of download languages. When this solution program is downloaded to a drive, the alarm texts of the download languages are included in the download. After the download, the desired language of these downloaded languages can be selected in the drive by the language selection parameter.
- Selection of the currently active language. The selected language must belong to the set of download languages. Texts of the selected language are now displayed in the textual items of the Alarm Manger window. If a textual item has no text defined in the selected language, its text in the default language (if such text exists) is displayed. All texts that are now written to the items of the Alarm Manager will belong to the selected language.

The name of the currently active language is displayed in the **Language** button.

Click the **Language** button. A dialog box is displayed:



Selection of the download languages

English is the default language, and it is always one of the download languages.

Other languages can be added to the set of download languages by the **Add language to the set of download languages** button.

Languages can be removed from the set of download languages by the **Remove language from the set of download languages** button.
The default language cannot be removed.

Finally, select one of the download languages and click **OK**.
The selected language will be the currently active language. See below.

Selection of the currently active language

Select the desired language from the set of download languages. Click **OK**.

Text in the selected language is now displayed in the textual items of the Alarm Manager window.
If a textual item has no text defined in the selected language, its text in the default language (if such text exists) is displayed.

Any text that is now entered to the items of the Alarm Manager will belong to the selected language.

Alarms

Every alarm contains two lines:

- Header line
- Value line

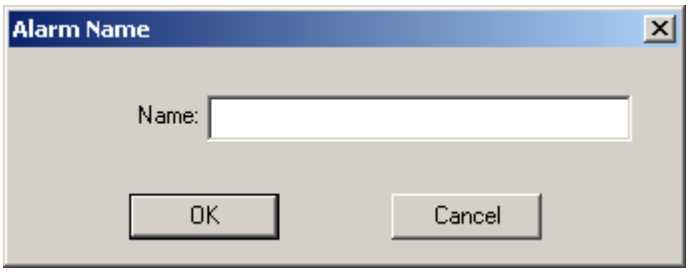
A new alarm is added by the **Add new** button in the header line of the alarm after which the new alarm will be added.

The alarm is deleted by right-clicking its **header line**. The first alarm can be deleted if it is the only alarm.

Value items in a value line are set or changed by left-clicking the desired **value area**.

Alarm name

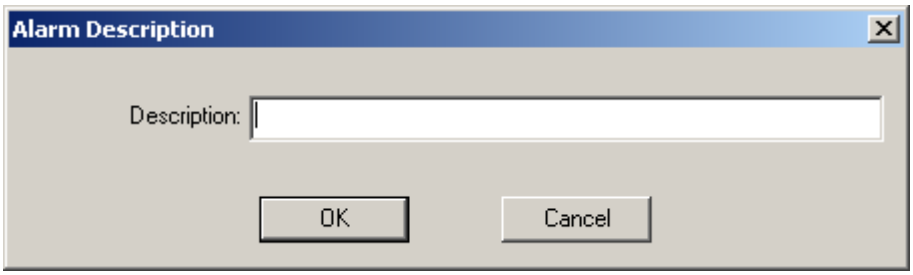
Alarm name is specified by left-clicking the alarm's **Alarm name** field. A dialog box is displayed:



Enter the desired alarm name. Click **OK**.

Description

Alarm description is specified by left-clicking the alarm's **Description** field. A dialog box is displayed:



Enter the desired alarm description. Click **OK**.

Appendix 5

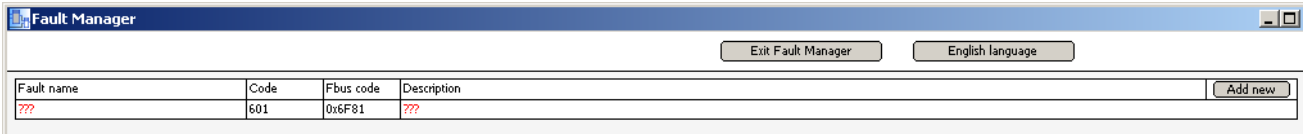
Appendix overview

This appendix describes the usage of the Fault Manager window.

Fault Manager

The Fault Manager window is opened by the menu command **Program > Fault Manager**.

If there are no user faults defined when the Fault Manager window is opened, there is one (undefined) fault in the window as follows:



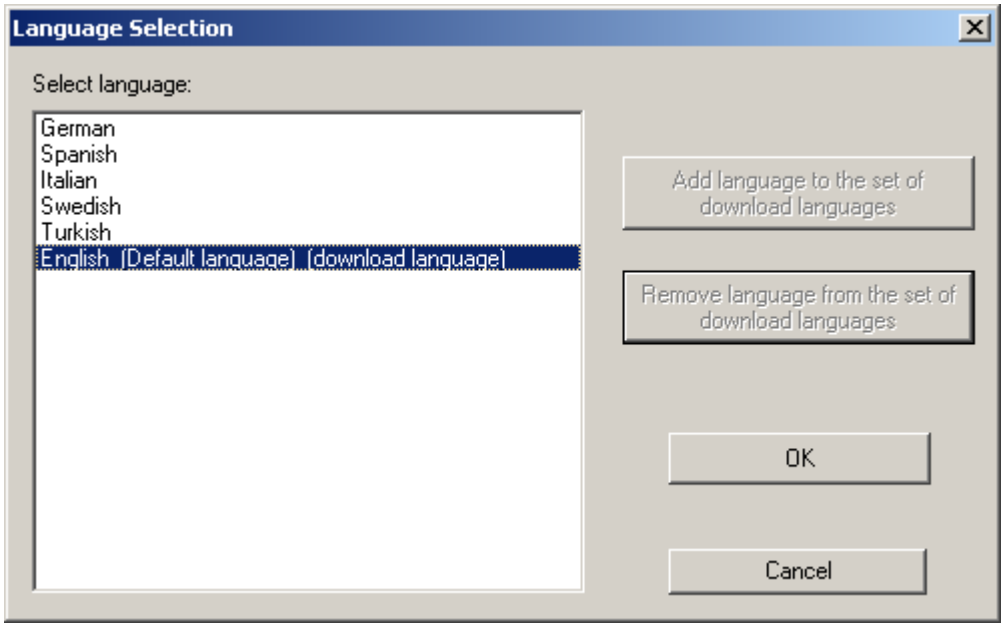
Languages of fault texts

The **Language** button at the top of the window is used for two purposes:

- Selection of the languages that belong to the set of download languages. When this solution program is downloaded to a drive, the fault texts of the download languages are included in the download. After the download, the desired language of these downloaded languages can be selected in the drive by the language selection parameter.
- Selection of the currently active language. The selected language must belong to the set of download languages. Texts of the selected language are now displayed in the textual items of the Fault Manger window. If a textual item has no text defined in the selected language, its text in the default language (if such text exists) is displayed. All texts that are now written to the items of the Fault Manager will belong to the selected language.

The name of the currently active language is displayed in the **Language** button.

Click the **Language** button. A dialog box is displayed:



Selection of the download languages

English is the default language, and it is always one of the download languages.

Other languages can be added to the set of download languages by the **Add language to the set of download languages** button.

Languages can be removed from the set of download languages by the **Remove language from the set of download languages** button.
The default language cannot be removed.

Finally, select one of the download languages and click **OK**.
The selected language will be the currently active language. See below.

Selection of the currently active language

Select the desired language from the set of download languages. Click **OK**.

Text in the selected language is now displayed in the textual items of the Fault Manager window.
If a textual item has no text defined in the selected language, its text in the default language (if such text exists) is displayed.

Any text that is now entered to the items of the Fault Manager will belong to the selected language.

Faults

Every fault contains two lines:

- Header line
- Value line

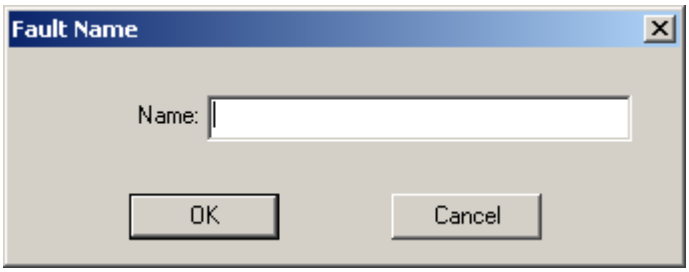
A new fault is added by the **Add new** button in the header line of the fault after which the new fault will be added.

The fault is deleted by right-clicking its **header line**. The first fault can be deleted if it is the only fault.

Value items in a value line are set or changed by left-clicking the desired **value area**.

Fault name

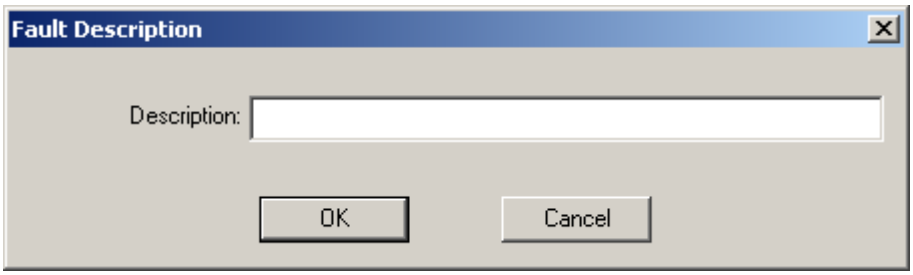
Fault name is specified by left-clicking the fault's **Fault name** field. A dialog box is displayed:



Enter the desired fault name. Click **OK**.

Description

Fault description is specified by left-clicking the fault's **Description** field. A dialog box is displayed:



Enter the desired fault description. Click **OK**.

Appendix 6

Appendix overview

This appendix describes the Structured Text language implementation of DriveSPC.

Structured Text language

In addition to block programming, custom circuits can be programmed by using the *Structured Text* (ST) programming language.

Standard IEC 61131-3 defines the Structured Text language.

The conformance of DriveSPC ST with this standard is described in the chapter *Compliance list of DriveSPC ST* in this Appendix.

The general structure of every custom circuit ST program must be as follows:

```
CUSTOM_CIRCUIT
(* Definitions of variables for external input pins (if any) *)
(* Definitions of variables for external output pins (if any) *)
(* Definitions of internal variables (if any) and
   definitions of function block instances (if any) *)
(* Executable program statements *)
END_CUSTOM_CIRCUIT
```

A program example is at the end of this Appendix.

Comments

User comments are delimited at the beginning and end by the special character combinations "`(*`" and "`*)`", respectively.

Nested comments are not allowed.

Example: `(* This is a comment *)`

Data types

Allowable data types are:

- *BOOL* (Boolean value `FALSE` (0) or `TRUE` (1))
- *INT* (16-bit signed integer value)
- *DINT* (32-bit signed integer value)
- *REAL* (32-bit real value. Value range is -32767.999 ... 32767.999)
- *REAL24* (32-bit real value. Value range is -127.999 … 127.999)

Note: The internal format of real values in DriveSPC ST is not the standard IEEE format but it is the format used with our function blocks. This results in the smaller value range specified above and greater granularity in the decimal parts of real values.

Constants (numeric literals)

A constant can be:

- *Integer number* (allowable range is -2147483648 ... 2147483647)
- *Real number* (allowable range is -32767.999 ... 32767.999)
- *Based number* (unsigned 32-bit binary, octal or hexadecimal value)
- *Boolean value* (`FALSE` (0) or `TRUE` (1))

Single underline characters inserted between the digits of a constant are not significant. No other use of underline characters in constants is allowed.

Integer numbers are represented in conventional decimal notation.

Real numbers are distinguished by having a decimal point.
An exponent indicates the integer power of ten by which the preceding number is to be multiplied to obtain the value represented.

Integer and real constants (and exponents) can contain a preceding sign (`+` or `-`).

Integer constants can also be represented in base 2, 8, or 16.
The format of a based number is `base#digits`. The base is in decimal notation.
For base 16, an extended set of digits consisting of the letters `A` through `F` is used, with the onventional significance of decimal 10 through 15, respectively.
Based numbers cannot contain a leading sign (`+` or `-`).

Boolean values are represented by integer constants with the value zero (`0`) or one (`1`), or the keywords `FALSE` or `TRUE`, respectively.

Examples:
```
12_345 -890        (* integer values *)
123.5 -19.0        (* real values *)
2#1000_1010        (* binary value *)
8#277              (* octal value *)
16#7_FFFF          (* hexadecimal value *)
FALSE TRUE         (* Boolean values 0 and 1 *)
```

Keywords

Keywords are unique combinations of characters used as individual syntactic elements of the ST language.

The keywords used by DriveSPC ST are listed in the chapter *Keywords of DriveSPC ST* in this Appendix.

The keywords must not be used for any other purpose (for example, variable names).

Keywords are case-insensitive.

Variables

User-defined variables provide a means of identifying data objects whose contents may change. A variable can be an external input or output pin, or an internal program data item.

A variable can be defined to be one of the data types listed in the chapter *Data types*.

A variable name is a string of letters, digits, and underline characters that begin with a letter or underline character.

The case of letters is insignificant but underlines are significant.

Multiple leading or multiple embedded underlines are not allowed.

Trailing underlines are not allowed.

The maximum length of variable names is 14 characters.

Examples: `x`, `Actual_speed`, `_Limit_value`, `MaxFrequency`

Block instances

Multiple, named instances (copies) of a function block in the Standard or Technology library can be created.

Each instance has an associated user-defined instance name.

An instance name is a string of letters, digits, and underline characters which begin with a letter or underline character.
The case of letters is insignificant, but underlines are significant.
Multiple leading or multiple embedded underlines are not allowed.
Trailing underlines are not allowed.

If a block requires a user-specified data type, the underline character and the name of the data type must be at the end of its instance name (e.g., `MyADD_DINT`).

The maximum length of instance names is 14 characters.

Examples: `ADD1_REAL`, `MyPID`

Definition statements

Variables and block instances to be used in a ST program must be defined in the beginning of the program before their usage in the executable statements of the program. (Executable statements are described in the chapter *Executable statements*).

The formats of definition statements are as follows:

```
VAR_INPUT  (* External input pin variables *)
  Variable definition lines
END_VAR

VAR_OUTPUT  (* External output pin variables *)
  Variable definition lines
END_VAR

VAR  (* Internal variables and block instances *)
  Variable and/or block instance definition lines
END_VAR
```

The format of a *variable definition line* is as follows:

```
VarName,VarName,...,VarName:DataType;
```

`DataType` must be one of the data types listed in the chapter *Data types*.

Variables defined with `VAR_OUTPUT` or `VAR` can be given an initial constant value in their *variable definition line* as follows:

```
VarName,VarName,...,VarName:DataType := InitValue;
```

The format of a *block instance definition line* is as follows:

```
InstName,InstName,...,InstName:BlockType;
```

`BlockType` must be one of the blocks in the Standard or Technology library.

Definitions of external input pin variables

Variables defined with `VAR_INPUT...END_VAR` are visible as input pins in the instances of this custom circuit at the main level of solution program.

Example:
```
VAR_INPUT
  Input1,Input2,Input3:REAL;
END_VAR
```

Definitions of external output pin variables

Variables defined with `VAR_OUTPUT...END_VAR` are visible as output pins in the instances of this custom circuit at the main level of solution program.

Example:
```
VAR_OUTPUT
  Output:REAL;
END_VAR
```

Definitions of internal variables and block instances

Internal variables and block instances of a custom circuit are defined with `VAR...END_VAR`.

Example:
```
VAR
  InterValue,Inter_value:REAL;
  ValueX:DINT := 10;
  PID_I1,PID_I2:PID;
  ADD1_DINT:ADD;
END_VAR
```

Expressions

An expression is a construct which, when evaluated, yields a value corresponding to one of the allowable data types.

Expressions are composed of operators and operands.

An operand can be:

- *Constant*
- *Variable*
- *Output pin of a block instance*
- *Another expression*

The allowable operators are:

- *Parenthesization* (highest precedence)
- *Negation* (-)
- *Complement* (NOT)
- *Exponentiation* (**)
- *Multiply* (*)
- *Divide* (/)
- *Modulo* (MOD)
- *Add* (+)
- *Subtract* (-)
- *Comparison* (<, >, <=, >=)
- *Equality* (=)
- *Inequality* (<>)
- *Boolean AND* (&)
- *Boolean AND* (AND)
- *Boolean Exclusive OR* (XOR)
- *Boolean OR* (OR) (lowest precedence)

The evaluation of an expression consists of applying the operators to the operands in a sequence defined by the operator precedence shown in the list above.

The operator with highest precedence in an expression is applied first, followed by the operator of next lower precedence, etc., until evaluation is complete.

Operators of equal precedence are applied as written in the expression from left to right.

When an operator has two operands, the leftmost operand is evaluated first.

Example: `(x + y) * z`

Executable statements

Executable statements must be after all the definition statements described in the chapter *Definition statements*.

Executable statements can be grouped as follows:

- *Assignment* statements
- *IF* statements
- *CASE* statements
- *FOR* statements
- *WHILE* statements
- *REPEAT* statements
- *EXIT* statements
- *RETURN* statements
- *Function block invocation* statements

All executable statements must be terminated with the semicolon character (`;`).

Assignment statement

The assignment statement replaces the current value of a variable by the result of evaluating an expression.

An assignment statement consists of a variable on the left-hand side, followed by the assignment operator "`:=`", followed by the expression to be evaluated.

Example:
```
Counter := Counter + 1;
```

IF statement

The IF statement specifies that a group of statements is to be executed only if the associated Boolean expression evaluates to the value 1 (true).

If the condition is false, then either no statement is to be executed, or the statement group following the `ELSE` keyword (or the `ELSIF` keyword if its associated Boolean condition is true) is to be executed.

Example:
```
IF a + b > 0 THEN
  VarX := ax + 5;
ELSIF t THEN
  VarX := bx + bz;
  y := 10;
ELSE
  K := K + 1;
END_IF;
```

CASE statement

The CASE statement consists of an expression which evaluates to a variable of type INT or DINT, and a list of statement groups, each group being labelled by one or more integer values or ranges of integer values, as applicable.

It specifies that the first group of statements, one of whose ranges contains the computed value of the selector, is executed.

If the value of the selector does not occur in a range of any case, the statement sequence following the keyword `ELSE` (if it occurs in the CASE statement) is executed.

Otherwise, none of the statement sequences is executed.

Example:
```
CASE sel OF
  1,4,5:
    a := b + c;
  2:
    x = Y * Z -10;
  3,6..10:
    val := -123.5;
  ELSE
    err := TRUE;
END_CASE;
```

FOR statement

The FOR statement indicates that a statement sequence is repeatedly executed, up to the `END_FOR` keyword, while a progression of values is assigned to the FOR loop control variable.

The control variable, initial value, and final value are expressions of the same integer type (INT or DINT) and must not be altered by any of the repeated statements.

The FOR statement increments the control variable up or down from an initial value to a final value in increments determined by the value of an expression; this value defaults to 1.

The test for the termination condition is made at the beginning of each iteration, so that the statement sequence is not executed if the initial value exceeds the final value.

Warning: **If a FOR loop takes too much execution time, the drive will fault after the download of the program.**

Example:

```
FOR xv := 0 TO 10 BY 2 DO
  k := xx / b;
END_FOR;
```

WHILE statement

The WHILE statement causes the sequence of statements up to the `END_WHILE` keyword to be executed repeatedly until the associated Boolean expression is false.

If the expression is initially false, then the group of statements is not executed at all.

Warning: **If a WHILE loop takes too much execution time, the drive will fault after the download of the program.**

Example:

```
WHILE X > 0 DO
  X := X - Z;
END_WHILE;
```

REPEAT statement

The REPEAT statement causes the sequence of statements up to the `UNTIL` keyword to be executed repeatedly (and at least once) until the associated Boolean condition is true.

Warning: **If a REPEAT loop takes too much execution time, the drive will fault after the download of the program**.

Example:

```
REPEAT
  var := var + 12;
UNTIL var >= 100
END_REPEAT;
```

EXIT statement

The EXIT statement can be used to terminate iterations before the termination condition is satisfied.

When the EXIT statement is located within nested iterative constructs, it exits from the innermost loop in which the `EXIT` is located, that is, control passes to the next statement after the first loop terminator (`END_FOR`, `END_WHILE`, or `END_REPEAT`) following the EXIT statement.

Example:

```
FOR xv := 0 TO 10 BY 2 DO
  b := xv + z;
  IF b > 100 THEN
    EXIT;
  END_IF;
  k := xx / b;
END_FOR;
```

RETURN statement

The RETURN statement provides an early exit from a program (for example, as the result of the evaluation of an IF statement).

Example:

```
IF err THEN
  RETURN;
END_IF;
```

Function block invocation statement

Function blocks of the Standard and Technology libraries can be invoked by a statement consisting of the user-specified instance name of the function block instance followed by a parenthesized list of arguments.

The arguments in the list are separated by commas.

The argument list has the form of a set of assignments of actual values to the pins:

- Assignments of values (expressions) to input pins using the ":=" operator.
- Assignments of the values of output pins to variables using the "=>" operator.

The order of arguments (pins) in this list is insignificant.

If an input pin is omitted from the list, its value is set to zero.

If desired, output pins can be omitted from the list.

Example: `ADD1_DINT (In1 := x, In2 := 100, Out => y);`

Usage of output pins of function blocks in expressions

Output pins of block instances can be used in expressions.

Example: `k1 := ADD1_DINT.Out + 12345;`

Compliance list of DriveSPC ST

Conformance of DriveSPC ST with the ST definition in the standard IEC 61131-3 (second edition) is described in the list below. Only the supported items of the standard are listed here. The numbers in the first column refer to the table items of the standard.

Table - No.	*Description*	*Remarks*
	Character set features	
1-2	Lower case letters	
1-3a	Number sign (#)	In base 2, 8 and 16 literals only
	Identifier features	Maximum length of identifiers is 14 characters
2-1	Upper case and numbers	
2-2	Upper and lower case, numbers, embedded underlines	
2-3	Upper and lower case, numbers, leading or embedded underlines	
	Comment feature	
3-1	Comments	
	Numeric literals	
4-1	Integer literals	Range -2147483648 ... 2147483647
4-2	Real literals	**Range** -32767.999 ... 32767.999
4-3	Real literals with exponents	**Range** -32767.999 ... 32767.999
4-4	Base 2 literals	32 bits
4-5	Base 8 literals	32 bit value
4-6	Base 16 literals	Range 0 … FFFFFFFF
4-7	Boolean zero and one	
4-8	Boolean FALSE and TRUE	
	Elementary data types	
10-1	BOOL	
10-3	INT	
10-4	DINT	
10-10	REAL	**Range** -32767.999 ... 32767.999
	Variable declaration	
VAR	Internal to organization unit	
VAR_INPUT	Externally supplied, not modifiable within organization unit	External input pin of custom circuit
VAR_OUTPUT	Supplied by organization unit to external entities	External output pin of custom circuit

	Variable type assignment	
17-5	Automatic memory allocation of symbolic variables	
18-5	Initialization of symbolic variables	
	Invocation of function blocks	
19a-1	Formal invocation type	
	Operators of the ST language	
55-1	Parenthesization	
55-3	Exponentiation (**)	
55-4	Negation (-)	
55-5	Complement (NOT)	
55-6	Multiply (*)	
55-7	Divide (/)	
55-8	Modulo (MOD)	
55-9	Add (+)	
55-10	Subtract (-)	
55-11	Comparison (<, >, <=, >=)	
55-12	Equality (=)	
55-13	Ineaquality (<>)	
55-14	Boolean AND (&)	
55-15	Boolean AND (AND)	
55-16	Boolean Exclusive OR (XOR)	
55-17	Boolean OR (OR)	
	ST language statements	
56-1	Assignment	
56-2	Function block invocation and FB output usage	Function blocks in the Standard and Technology libraries can be used
56-3	RETURN	
56-4	IF	
56-5	CASE	
56-6	FOR	
56-7	WHILE	
56-8	REPEAT	
56-9	EXIT	
56-10	Empty statement	

Keywords of DriveSPC ST

AND

CASE...OF...ELSE...END_CASE

CUSTOM_CIRCUIT...END_CUSTOM_CIRCUIT

EXIT

FALSE

FOR...TO...BY...DO...END_FOR

IF...THEN...ELSIF...ELSE...END_IF

MOD

NOT

OR

REPEAT...UNTIL...END_REPEAT

RETURN

TRUE

VAR...END_VAR

VAR_INPUT...END_VAR

VAR_OUTPUT...END_VAR

WHILE...DO...END_WHILE

XOR

Names of the allowable data types

Names of the function blocks in the Standard and Technology libraries

Program example

```
CUSTOM_CIRCUIT

      (* Define external input pins of custom circuit *)
VAR_INPUT
  Bits,BitSelect:DINT;
  InValue1,InValue2:REAL;
END_VAR

      (* Define external output pins of custom circuit *)
VAR_OUTPUT
  SelectedBit:BOOL;
  OutputValue:REAL;
END_VAR

     (* Define block instance of block BGET *)
     (* BGET block instance requires data type *)
VAR
  BitGet_DINT:BGET;  (* Data type of our instance is DINT *)
END_VAR

      (* Get bit and put its value to SelectedBit *)
BitGet_DINT (BITNR := BitSelect, I := Bits, O => SelectedBit);

      (* Calculate the value of OutputValue *)
IF BitGet_DINT.O = TRUE THEN
  OutputValue := (InValue1 + 100.0) * InValue2;
ELSE
  OutputValue := InValue1;
END_IF;

END_CUSTOM_CIRCUIT
```

Appendix 7

Appendix overview

This appendix describes the additional functionality of the Pro version of DriveSPC.

Pro version

The following additional features are available in the Pro version of DriveSPC:

- Function blocks can be added to the faster time levels of firmware blocks. ***Warning:*** **Only a very limited number of blocks can be added to these time levels. If the execution time of the added blocks is too long, the drive will fault after the download of the program.**
- Time lengths of time levels of firmware blocks can be changed. ***Warning:*** **If you specify too short time level times, the drive will fault after the download of the program.**
- Values and connections of input pins of non-firmware blocks can be changed in the *On-Line* mode as in the *Off-Line* mode.

Note that the changed *On-Line* pin values/connections remain in the drive until the drive is powered down. The original, programmed values/connections of the changed pins are restored in the next power-up of the drive.
To change the actual *On-Line* pin values/connections currently on the screen to programmed values of the drive:

1. Save the current program with actual values by selecting **File > Save**
2. Go back to the *Off-Line* mode by selecting **Drive > Off-Line**.
3. Open the saved program file by selecting **File > Open**
4. Download the program to the drive by selecting **Drive > Download Program**

3AFE 68836590 REV K EN
EFFECTIVE: 1.6.2009

ABB Oy
AC Drives
P.O.Box 184
FIN-00381 HELSINKI
FINLAND
Telephone + 358 10 22 2000
Fax + 358 10 22 22681
Internet http://www.abb.com