

Rexroth Rho 4 BAPS plus

1070072187
Edition 05

Software manual



Title Rexroth Rho 4
BAPS plus

Type of Documentation Software manual

Document Typecode DOK-RHO*4*-BAPSP*SOFTH-PR05-EN-P

Purpose of Documentation The present manual informs about:

- the use of the graphical programming system BAPS plus.

Record of Revisions

Description	Release Date	Notes
DOK-RHO*4*-BAPSP*SOFTH-PR04-EN-P	10.2003	Valid from VO07
DOK-RHO*4*-BAPSP*SOFTH-PR05-EN-P	01.2005	Valid from VO08

Copyright © Bosch Rexroth AG, 1998 – 2005
Copying this document, giving it to others and the use or communication of the contents thereof without express authority, are forbidden. Offenders are liable for the payment of damages. All rights are reserved in the event of the grant of a patent or the registration of a utility model or design (DIN 34–1).

Validity The specified data is for product description purposes only and may not be deemed to be guaranteed unless expressly confirmed in the contract. All rights are reserved with respect to the content of this documentation and the availability of the product.

Published by Bosch Rexroth AG
Postfach 11 62
D-64701 Erbach
Berliner Straße 25
D-64711 Erbach
Tel.: +49 (0) 60 62/78-0
Fax: +49 (0) 60 62/78-4 28
Abt.: BRC/ESH (KW)

Overview of all manuals

Overview of all manuals

Manual	Contents
Connection conditions Rho 4.0	2 System overview
	3 Installation
	4 Electrical connection
	5 Interfaces
	6 LED display
	7 Maintenance and replacement
	8 Order numbers
System description Rho 4.0	2 System overview
	3 Structure of the rho4.0
	4 PCLrho4.0
	5 CAN-Bus peripheral unit
	6 SERCOS interface
	7 Software
	8 File management
Connection conditions Rho4.1, Rho 4.1/IPC300	2 System overview
	3 Security functions
	4 Installation
	5 Electrical connection
	6 Interfaces
	7 LED display
	8 Maintenance and replacement
	9 Software
	10 Order numbers
Connection conditions Rho 4.1/BT155, Rho 4.1/BT155T, Rho 4.1/BT205	2 System overview
	3 Security functions
	4 Installation
	5 Electrical Connections
	6 Interfaces
	7 Display and Operating Controls
	8 Maintenance and Replacemant
	9 Software
	10 Order numbers
System description Rho 4.1	2 Structure of the rho4.1
	3 PCL
	4 CAN-Bus peripheral unit
	5 SERCOS interface

Overview of all manuals

Manual	Contents
	6 Software
	7 File management
	8 Scope of the rho4.1 Software

Manual	Contents
Control functions	2 Survey of special functions
	3 Accurate position switching
	4 Setting the machine position
	5 Calling operating system functions
	6 Parameterization of the belt characteristic
	7 Selecting a point-file
	8 Mirroring
	9 Belt type
	10 System date and time
	11 System counter
	12 WC main range
	13 Setting the belt counter
	14 Recording of reference path
	15 Flying measurement (rho4.1 only)
	16 MOVE_FILE
	17 Setting the block preparation
	18 Exception-Handling
	19 Belt counter current value
	20 Automatic velocity adjustment for PTP movements
	21 Belt-synchronous working area belt kind 4
	22 Current belt speed
	23 Changing the belt simulation speed
	24 General functions
	25 Process-oriented functions
	26 BAPS3 keywords
	Machine parameters
3 Application of the machine parameters	
4 General system parameters	
5 Speeds	
6 Positions	
7 Kinematic parameters	

Overview of all manuals

Manual	Contents
	8 Measuring system parameters
	9 Belt parameters
	10 Drive parameters Servodyn-GC
	11 Drive parameter Servodyn-D
	12 Table of parameters
Manual	Contents
BAPS3 Programming manual	2 Program structure
	3 Constants
	4 Variables
	5 Program control
	6 Value assignments and combinations
	7 Functions
	8 Movement statement
	9 Write/read functions
	10 BAPS3 keywords
	BAPS3 Short description
3 Constants and variables	
4 Program structure	
5 Value assignments and combinations	
6 Standard functions	
7 Movements and speeds	
8 Belt synchronous	
9 Workspace limitation	
10 Write/read functions	
11 Special functions	
12 Library functions	
13 Fix files	
14 BAPS3 keywords	
Signal descriptions	
	3 Signal description of PCL inputs
	4 Signal description of PCL outputs
Status messages and warnings	2 rho4 status messages
	3 Warnings
	4 CANopen error codes
ROPS4/Online	2 General information
	3 Activation and functions of Online
	4 The function key box

Overview of all manuals

Manual	Contents
	5 Function key assignment
	6 The marker box
	7 File ROPS4WIN.ini
	8 Selection of a file
	9 TCP/IP settings for ROPS4
Manual	Contents
DLL library	2 Library functions
	3 Calling library functions in BAPS
	4 Block structure of the rho4.1
	5 Library server
	6 Application development
	7 rho4 library functions
	8 Variable access per DLL
	PHG2000
3 PHG2000 system variables	
4 Selection of PHG functions	
5 Info function of the PHG	
6 Controlling the PHG2000 output	
7 Define/Teach	
8 SRCAN functions	
9 File and User Memory Functions	
10 File list	
11 Process info	
12 Restoring the PGH display	
13 Variable assignment of PHG keys	
14 Select point file and point name	
15 BDT editor	
Connection conditions Rho 4.1/IPC 40.2	
	3 Security Functions
	4 Installation
	5 Eelectrical Connections
	6 Interface Ports & Connectors
	7 Display- and Operating Components
	8 Maintenance and Replacement
	9 Software
	10 Ordering Informations

Overview of all manuals

Manual	Contents
DDE-Server	2 Introduction
	3 Hardware and Software
	4 Operation
	5 Items of Server 4
	6 Scope of function

Overview of all manuals

Notes:

Contents

Contents

	Page
1	Safety Instructions 1–1
1.1	Intended use 1–1
1.2	Qualified personnel 1–2
1.3	Safety markings on products 1–3
1.4	Safety instructions in this manual 1–4
1.5	Safety instructions for the described product 1–5
1.6	Documentation, software release and trademarks 1–7
2	General information 2–1
3	Fundamental principles 3–1
3.1	Installation 3–1
3.1.1	Directory structure 3–2
3.1.2	Data types 3–3
3.1.3	First steps with BAPSplus 3–5
3.2	The operating surface 3–7
3.2.1	Optional window 3–20
4	Elementary functions 4–1
4.1	Start BAPSplus 4–1
4.2	Main menu items 4–3
4.2.1	Main menu item 'File' 4–3
4.2.2	Main menu item 'Edit' 4–10
4.2.3	Main menu item 'Item' 4–14
4.2.4	Main menu item 'View' 4–17
4.2.5	Main menu item 'Debug' 4–19
4.2.6	Main menu item 'Library' 4–25
4.2.7	Main menu item 'Coupling' 4–28
4.2.8	Main menu item 'Extras' 4–30
4.2.9	Main menu item 'Options' 4–33
4.2.10	Main menu item 'Tool' 4–38
4.2.11	Main menu item '?' (Help) 4–39
5	Expanded Functions 5–1
5.1	Presettings and program data 5–1
5.1.1	Default kinematics 5–1
5.1.2	Program declarations 5–3
5.1.3	Definition of constants 5–5
5.1.4	Declaration of type 5–7
5.1.5	Declaration of variables 5–10
5.1.6	Define signals 5–14
5.1.7	Define subroutines 5–16
5.1.8	Display change history 5–19
5.2	Incremental Compiler 5–20

Contents

5.3	Test programs	5-21
5.4	Working with program masters	5-21
5.5	Working with libraries	5-22
5.6	Working with the icon bar	5-23
5.6.1	Icon grammar	5-23
5.7	Working with new commands	5-26
5.8	Use export library files	5-31
5.9	Working with tools	5-32
6	Data Input Manager	6-1
6.1	Function mode	6-1
7	BAPSplus Recompiler	7-1
7.1	Function description	7-1
7.2	Calling ReCompiler	7-2
7.3	Import BAPS Sourcecode	7-5
7.4	Functions	7-7
7.4.1	File name and file version of the BAPSplus program	7-8
7.4.2	Language version of the BAPS source text	7-8
7.4.3	Log file	7-8
7.4.4	Settings of the ReCompiler	7-13
7.5	Special cases	7-15
7.6	Example: BAPS source text import	7-18
8	Setting possibilities of the INI file	8-1
8.1	TOOLS configuration	8-1
8.2	Import configuration	8-2
8.3	BAPSplus setting	8-3
8.4	Settings of the incremental compiler	8-6
8.5	Other groups	8-6
9	Example	9-1
10	Shortcuts and tool bars	10-1
A	Appendix	A-1
A.1	Abbreviations	A-1
A.2	Index	A-2

1 Safety Instructions

Please read this manual before you startup the rho4.
Store this manual in a place to which all users have access at any time.

1.1 Intended use


This instruction manual presents a comprehensive set of instructions and information required for the standard operation of the described products. The described products are used for the purpose of operating with a robot control rho4.

The products described

- have been developed, manufactured, tested and documented in compliance with the safety standards. These products normally pose no danger to persons or property if they are used in accordance with the handling stipulations and safety notes prescribed for their configuration, mounting, and proper operation.
- comply with the requirements of
 - the EMC Directives (89/336/EEC, 93/68/EEC and 93/44/EEC)
 - the Low-Voltage Directive (73/23/EEC)
 - the harmonized standards EN 50081-2 and EN 50082-2
- are designed for operation in industrial environments, i.e.
 - no direct connection to public low-voltage power supply,
 - connection to the medium- or high-voltage system via a transformer.

The following applies for application within a personal residence, in business areas, on retail premises or in a small-industry setting:

- Installation in a control cabinet or housing with high shield attenuation.
- Cables that exit the screened area must be provided with filtering or screening measures.
- The user will be required to obtain a single operating license issued by the appropriate national authority or approval body. In Germany, this is the Federal Institute for Posts and Telecommunications, and/or its local branch offices.

 **This is a Class A device. In a residential area, this device may cause radio interference. In such case, the user may be required to introduce suitable countermeasures, and to bear the cost of the same.**

The faultless, safe functioning of the product requires proper transport, storage, erection and installation as well as careful operation.

Safety Instructions

1.2 Qualified personnel

The requirements as to qualified personnel depend on the qualification profiles described by ZVEI (central association of the electrical industry) and VDMA (association of German machine and plant builders) in:

Weiterbildung in der Automatisierungstechnik

edited by: ZVEI and VDMA

MaschinenbauVerlag

Postfach 71 08 64

D-60498 Frankfurt.

The present manual is designed for RC technicians. They need special knowledge on handling and programming robots.

Interventions in the hardware and software of our products, unless described otherwise in this manual, are reserved to specialized Rexroth personnel.

Tampering with the hardware or software, ignoring warning signs attached to the components, or non-compliance with the warning notes given in this manual may result in serious bodily injury or damage to property.

Only electrotechnicians as recognized under IEC 60947-1 (modified) who are familiar with the contents of this manual may install and service the products described.

Such personnel are

- those who, being well trained and experienced in their field and familiar with the relevant norms, are able to analyze the jobs being carried out and recognize any hazards which may have arisen.
- those who have acquired the same amount of expert knowledge through years of experience that would normally be acquired through formal technical training.

With regard to the foregoing, please note our comprehensive range of training courses. Please visit our website at

<http://www.boschrexroth.com>

for the latest information concerning training courses, teachware and training systems. Personal information is available from our Didactic Center Erbach,

Telephone: (+49) (0) 60 62 78-600.

Safety Instructions

1.3 Safety markings on products

Warning of dangerous electrical voltage!



Warning of danger caused by batteries!



Electrostatically sensitive components!



Warning of hazardous light emissions
(optical fiber cable emissions)!



Disconnect mains power before opening!



Lug for connecting PE conductor only!



Functional earthing or low-noise earth only!



Connection of shield conductor only

Safety Instructions

1.4 Safety instructions in this manual



DANGEROUS ELECTRICAL VOLTAGE

This symbol is used to warn of a **dangerous electrical voltage**. The failure to observe the instructions in this manual in whole or in part may result in **personal injury**.



DANGER

This symbol is used wherever insufficient or lacking compliance with instructions may result in **personal injury**.



CAUTION

This symbol is used wherever insufficient or lacking compliance with instructions may result in **damage to equipment or data files**.

☞ This symbol is used to draw the user's attention to special circumstances.

★ This symbol is used if user activities are required.

Safety Instructions

1.5 Safety instructions for the described product**DANGER**

Danger of life through inadequate EMERGENCY-STOP devices! EMERGENCY-STOP devices must be active and within reach in all system modes. Releasing an EMERGENCY-STOP device must not result in an uncontrolled restart of the system! First check the EMERGENCY-STOP circuit, then switch the system on!

**DANGER**

**Danger for persons and equipment!
Test every new program before starting up a system!**

**DANGER**

**Retrofits or modifications may adversely affect the safety of the products described!
The consequences may include severe injury, damage to equipment, or environmental hazards. Possible retrofits or modifications to the system using third-party equipment therefore have to be approved by Rexroth.**

**DANGER**

Do not look directly into the LEDs in the optical fiber connection. Due to their high output, this may result in eye injuries. When the inverter is switched on, do not look into the LED or the open end of a short connected lead.

**DANGEROUS ELECTRICAL VOLTAGE**

Unless described otherwise, maintenance works must be performed on inactive systems! The system must be protected against unauthorized or accidental reclosing.

Measuring or test activities on the live system are reserved to qualified electrical personnel!

Safety Instructions

**CAUTION****Danger to the module!**

Do not insert or remove the module while the controller is switched ON! This may destroy the module. Prior to inserting or removing the module, switch OFF or remove the power supply module of the controller, external power supply and signal voltage!

**CAUTION****use only spare parts approved by Rexroth!****CAUTION****Danger to the module!**

All ESD protection measures must be observed when using the module! Prevent electrostatic discharges!

The following protective measures must be observed for modules and components sensitive to electrostatic discharge (ESD)!

- Personnel responsible for storage, transport, and handling must have training in ESD protection.
- ESD-sensitive components must be stored and transported in the prescribed protective packaging.
- ESD-sensitive components may only be handled at special ESD-workplaces.
- Personnel, working surfaces, as well as all equipment and tools which may come into contact with ESD-sensitive components must have the same potential (e.g. by grounding).
- Wear an approved grounding bracelet. The grounding bracelet must be connected with the working surface through a cable with an integrated 1 MΩ resistor.
- ESD-sensitive components may by no means come into contact with chargeable objects, including most plastic materials.
- When ESD-sensitive components are installed in or removed from equipment, the equipment must be de-energized.

Safety Instructions

1.6 Documentation, software release and trademarks

Documentation

The present manual provides information on the use of the graphical programming system BAPS plus.

Overview of available documentation	Part no.	
	German	English
Rho 4.0 Connectivity Manual	1070 072 364	1070 072 365
Rho 4.0 System description	1070 072 366	1070 072 367
Rho 4.1/IPC 40.2 Connectivity Manual	R911308219	R911308220
Rho 4.1/BT155, Rho 4.1/BT155T, Rho 4.1/BT205 Connectivity manual	1070 072 362	1070 072 363
Rho 4.1, Rho 4.1/IPC300 Connectivity manual	1070 072 360	1070 072 361
Control panels BF2xxT/BF3xxT, connection	1070 073 814	1070 073 824
Rho 4.1 System description	1070 072 434	1070 072 185
ROPS4/Online	1070 072 423	1070 072 180
BAPS plus	1070 072 422	1070 072 187
BAPS3 Short description	1070 072 412	1070 072 177
BAPS3 Programming manual	1070 072 413	1070 072 178
Control functions	1070 072 420	1070 072 179
Signal descriptions	1070 072 415	1070 072 182
Status messages and warnings	1070 072 417	1070 072 181
Machine parameters	1070 072 414	1070 072 175
PHG2000	1070 072 421	1070 072 183
DDE-Server 4	1070 072 433	1070 072 184
DLL-Library	1070 072 418	1070 072 176
Rho 4 available documentation on CD ROM	1070 086 145	1070 086 145

 **In this manual the floppy disk drive always uses drive letter A:, and the hard disk drive always uses drive letter C:.**

Special keys or key combinations are shown enclosed in pointed brackets:

- Named keys: e.g., <Enter>, <PgUp>,
- Key combinations (pressed simultaneously): e.g., <Ctrl> + <PgUp>

Safety Instructions

Release

 **This manual refers to the following versions:**

Hardware version: rho4

Software release: ROPS4

Trademarks

All trademarks of software installed on Rexroth products upon delivery are the property of the respective manufacturer.

Upon delivery, all installed software is copyright-protected. The software may only be reproduced with the approval of Rexroth or in accordance with the license agreement of the respective manufacturer.

MS-DOS® and Windows™ are registered trademarks of Microsoft Corporation.

PROFIBUS® is a registered trademark of the PROFIBUS Nutzerorganisation e.V. (user organization).

MOBY® is a registered trademark of Siemens AG.

AS-I® is a registered trademark of AS-International Association.

SERCOS interface™ is a registered trademark of Interessengemeinschaft SERCOS interface e.V. (Joint VDW/ZVEI Working Committee).

INTERBUS-S® is a registered trade mark of Phoenix Contact.

DeviceNet® is a registered trade mark (TM) of ODVA (Open DeviceNet Vendor Association, Inc.).

General information

2 General information

This chapter briefly summarizes the new possibilities and functions of the graphical programming system BAPS plus.

Functional features

BAPS plus is a programming system which releases its user largely from learning a programming language and its syntax. The structured and systematic creation and updating of robot programs is facilitated considerably. The graphical Windows user interface provides the user furthermore with a working environment he is used to and which is easy to handle. To do so, the programmer compiles the program in form of a **Program Flowchart (PFC)** from symbols, so-called icons. The corresponding BAPS program text is generated automatically in the background and thus syntactically without any mistakes.

To structure the program, the programmer can summarize any number of commands in one element (formation of blocks). The elements summarized in this way are then set-up in a new level under the block element. This program structure with ever deeper sublevels at will supports a program creation according to the top-down methodology.

The entry of all parameter for the program commands (e. g. position description for the MOVE command) is made in command-specific entry masks, which offer the user all possibilities of a BAPS command without confronting him with the actual syntax. The error possibilities are thus reduced. This data is checked by an incremental translator after the closing of the corresponding entry mask, and any errors detected will be communicated to the user. Program parameters, variables, sub-routines, etc. are also entered in a corresponding pop-up window. The error possibilities are thus minimized, while achieving a maximum independence from the language. This has been taken into account by using graphical symbols in the flowcharts as symbols for the commands and functions.

Apart from using the contained standard program elements, the user can expand the system at will by self-established functions.

The ROPS4 'Online' module also permits a remote debugging of the established programs on the control. When doing so, it is possible to display and edit variables; also the use of breakpoints is possible.

The programming system also permits, without any additional work, the automatic generation of the program documentation in form of printed program flowcharts and .qll programs.

General information

Creation of application-specific operating surfaces

BAPSPPlus contains functions permitting an easy creation of customer-specific operating surfaces for the entry of parameters and/or the teach-in of new robot movements. The appearance, the functionalities of the operating surface and the connection with the robot program can to a large extent be freely defined by the user.

The connection between the values entered on the surface by the end-user and the corresponding RC program is made by the system automatically. A programming of the surface according to the common understanding is therefore no longer required. The programs created in this way are executed under the control of the Data Input Manager (DIM).

System requirements

To ensure an impeccable function of the graphical programming system, the following requirements must be met:

- Hardware (monitor, CPU, memory, disk space, etc.): The programming system can be run on a commercial PC, as used for normal operations. The video hardware must support a resolution of 1024 pixels x 768 pixels (for the graphic editor) resp. 640 pixels x 480 pixels (for the Data Input Manager) with minimum 16 colors. For the operation of the graphic editor a mouse is required, while the Data Input Manager can be fully operated with the keyboard.
- Operating system: Windows95 or WindowsNT is required as operating system.
- For the full utilisation of all functions offered, an additional software is required. Additional software: Workshop by Borland, AppStudio by Microsoft, etc. If e. g. own symbols or dialogs are to be created, a suitable graphic program or a dialog editor has to be used.

 **The standard scope of supply does not include the additional software.**

Fundamental principles

3 Fundamental principles

This chapter informs about the basic knowledge on functions, procedures and skills required to start working with the graphical programming system BAPSplus.

3.1 Installation

BAPSplus is part of the **Robot Offline Programming System ROPS4**.

 **BAPSplus does not support long file names. File names must correspond to the 8.3 – Convention. This also applies to the installation path.**

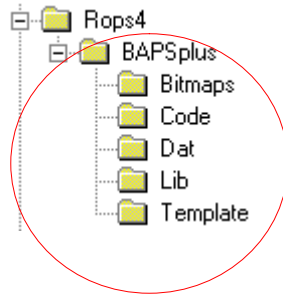
For the communication with the control, the Bosch DDE Server must be installed. Only in combination with the DDE Server the comprehensive test possibilities of the graphical programming system BAPSplus can fully be used.

The loading of files onto the control and/or the backup of this data (see section 4.2.7) onto a PC also calls for a correct installation of the DDE Server.

Fundamental principles

3.1.1 Directory structure

During the installation – proceeding from the selected program file (installation directory) – the following subdirectories are created:




- Program directory (BAPsplus)

In this directory, the application as well as all files required for the execution are saved. Also all configuration files are saved here.

- Library files (Lib)

The existing libraries (file extension .bib) and their configuration files are included here. If new libraries are to be created (see 'Create new file', page 4-3), they have to be saved in the 'Lib' directory.

 **The name 'Lib' is assigned during the installation, yet can be changed subsequently. Please make sure that the corresponding entry under 'Options/Directories...' is also adapted, see 'Directories...', page 4-35.**

- User icons (here Bitmaps)

The start screen of BAPsplus, OEMINFO.bmp, is placed here. This screen can be modified or replaced to realize your own start screens. Your own icons for new commands are also placed here, see chapter 6.

Fundamental principles

- Code directory (Code)

In the code directory, the automatically generated BAPS source programs (file extension .qll) and the executable rho4 files generated with the compilation of these programs are also placed here.

☞ **The name 'Code' is assigned during the installation, yet can be changed subsequently. Please make sure that also the corresponding entry under 'Options/Directories...' is adapted, see 'Directories...', page 4–35.**

- File directory (Dat)

The BAPSplus files (file extension .opd) and tool files possibly generated are saved here, see section 5.9.

☞ **The name 'Dat' is assigned during the installation, yet can be changed subsequently. Please make sure that also the corresponding entry under 'Options/Directories...' is adapted, see 'Directories...', page 4–35.**

- Master directory (Template)

Program masks, so-called templates or program masters, have to be stored in the 'Template' directory. BAPSplus offers all templates contained in this directory for selection when a new file is created (see section 5.4.).

☞ **The name 'Templates' can only be changed via an entry in the file 'BAPS-plus.ini', see 'TemplateDir', page 8–5.**

3.1.2 Data types

During the work with BAPSplus, the following data is created:

- Program files

Elementary file type of BAPSplus representing the basis of the BAPS program to be generated. The file extension is '.opd'.

- Library files

Basis for the expendability of BAPSplus. The file extension is '.bib'. New BAPS commands are generated and stored here. see section 5.5. By integrating a library, the functions contained in this are made available to the user.

- Export library files

Export library files serve the exchange of programs between several computers resp. users, see section 5.8. The file extension is '.opx'.

Fundamental principles

- Configuration files

Configuration files serve the storage of adjustments made once, (e. g. files edited last, window position, window size etc). The file extension is '.cfg'.

- Initialization files

The function of the initialization files is the default setting of the programming system. The file extension is '.ini'.

- Default kinematics file

In this file, the kinematics defined as standard are stored, see section 5.1.1. The file name is 'DEFAULT.okd'.

- Default tool file

In this file, the tools defined as standard are stored, see section 5.9. The file name is 'DEFAULT.owd'.

- Tool file

In the tool file the data of various grippers and/or tools are placed, see section 5.9. The file extension is '.dat'.

- Source files

The source files are generated automatically through BAPSplus and serve the BAPS compiler as input for the generation of the executable rho4 programs. The file name corresponds to that of the program file, the file extension is '.qll'.

- Error files

Errors and warnings recognized during the compilation are stored in the error file. The file name corresponds to that of the source file, the file extension is '.err'.

- Point files

The point files are generated by the BAPS compiler, provided that the source program can be compiled without any errors and contains teach points. The file name corresponds to that of the program file, the file extension is '.pkt'.

- Symbol files

The symbol files are generated by the BAPS compiler provided that the source program can be translated without any errors and the test information is not deactivated. Symbol files are required for the program test to be carried out later. The file name corresponds to that of the program file, the file extension is '.sym'.

- rho4 program files

These are the programs that can be run on the rho4. They are generated by the BAPS compiler, provided that the source program can be compiled without any errors. The file name corresponds to that of the program file, the file extension is '.ird'.

Fundamental principles

3.1.3 First steps with BAPSplus

This section gives a short instruction of the steps required to get from the program start to a program ready to be used.

Starting situation

The work surface of the program flowchart (PFC) only shows the BEGIN and the END element. They represent the skeleton of the program to be developed.

The corresponding program adjustments, such as number of kinematics, number of axes, used tools, etc., are determined by default files. The adjustments can be displayed and changed within the parameter dialog belonging to the BEGIN element. This dialog appears when the BEGIN element is clicked with the right-hand mouse key and the 'Parameter...' entry is selected in the displayed context menu.

The parameter dialog furthermore enables to enter or edit program parameters, constants, variables, signals and functions. The editor list of the current program can also be looked at here.

Insertion of symbols

At the left-hand rim of the main window there is the so-called icon bar, also called symbol bar. It contains the commands that are available and is divided into several columns, but only one is visible.

The desired column can be activated via the list box at the top rim, by scrolling with the scroll bar at the bottom rim or through the context menu of the symbol bar.

 **The field above the list box contains the four most frequently used symbols. They permit a quick access, without having to change the columns first.**

By clicking with the mouse, the desired symbol is inserted into the PFC under the current element (with yellow rim). The BAPS program text is then generated automatically in the background.

 **By selecting the menu item 'View/View BAPS Sourcecode', the BAPS program text can be shown in a separate window.**

Fundamental principles

Entry of instruction parameters

The symbols in the program flowchart are provided with context menus. The most important menu item in them is the entry 'Parameter...' It opens, if available, the input dialog for the current command. Here, the desired data are entered or they are selected from the list boxes available.

- ☞ **If the <Shift> key is pressed during the selection of a command from the icon bar, the input dialog is opened automatically after the insertion.**

After having completed the dialog by selecting 'OK', the entered data are checked automatically. If an error occurs in the meantime, it is displayed in the status line. The dialog is opened again and the faulty input field selected directly.

- ☞ **The automatic check can be switched on or off via the menu point 'Options/Check BAPS Sourcecode'.**

Structuring of program

If a program consists of many symbols, it can no longer be fully seen in the PFC. BAPSplus offers the possibility of grouping any number of commands to one single element (block generation).

The grouped elements are then arranged in a new level under the block element. By structuring the program in this way with ever deeper sublevels, a programm creation according to the top-down methodology is supported.

To display a sublevel, a double-click has to be made on the associated block element. The return to the next higher level is made by a double-click on a free space in the PFC.

- ☞ **If the block element is the active element, it is sufficient to press the <Enter> key to change into the sublevel; the return is made with the <Backspace>.**

Test program

If the program is completed, it can be tested in the rho4 control. For this purpose, the menu item 'Debug' with its submenu items is available. The test mode is activated by the menu item 'Debug – Select program'; alternatively <F8> can be used.

Fundamental principles

The program is then loaded into the control and prepared for its execution. With the menu point 'Debug – Start/continue program', it is executed in the control. As soon as the program runs, the program code window is opened and the current program line is marked therein. The program execution can also be followed simultaneously in the program flowchart.

For a more detailed diagnosis, it is possible to set breakpoints in the program and to display and change variables. The test mode can at any time be aborted by selecting 'Debug – Stop Debugging'. It ends automatically when the end of program has been reached.

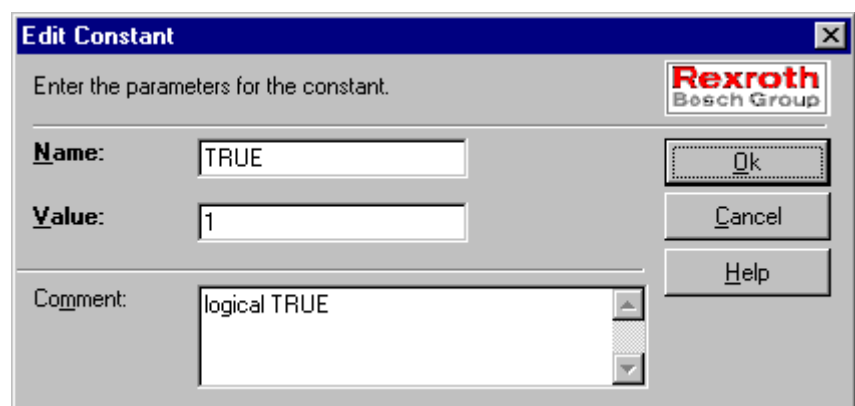
Program documentation

After a successful program test, the documentation can be printed out via 'File – Print...'. In the corresponding dialog it is possible to select at first which program parts are to be printed in which way. Also an adjustment of the margins, header space, scaling factor etc. is possible.

3.2 The operating surface

The operating surface of the programming system corresponds to that of other Windows applications:

- Operating elements (buttons, menu items etc.) that are not available at the moment, are deactivated, i. e. presented in grey. If it is possible to use them again, they will be reactivated.
- In the various input dialogs of the programming system, the elements to be filled-in in any case are marked in 'Bold' letters. In the following illustration, these are the elements 'Name' and 'Value'. The elements in standard typeface do not need to be entered in any case. They support, however, in most cases the understanding of the program, or represent optional components.



- The graphical programming system supports the German and the English languages.

Fundamental principles

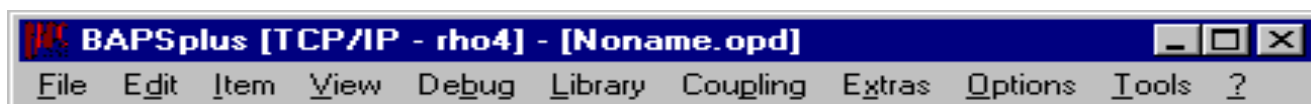
- The graphical control elements support so-called Win- or QuickTips (small information windows appearing when the mouse pointer remains for a short while on the corresponding element).

The header

The header contains the name of the application (BAPS plus), the type of coupling, the name of the target control and the name of the currently loaded program.

The menu bar

The menu bar permits the access to the various menu items with their functions, see chapter 4 'Elementary functions' and 5 'Expanded functions'.



The following tables list the individual menu items. For further information, please refer to the corresponding chapter.

Fundamental principles

Main menu item 'File'

Menu item	Signification
New	Create new file When selecting this item, a submenu containing the following items appears: <ul style="list-style-type: none"> Standard Generates a new program file with the current standard defaults Library Generates a new library Templates Generates a new program file, based on a template from the master directory, see section 5.4. Import BAPS source-code Adopt source text from existing BAPS-QLL program (see section 'Recompiler')
Open...	Load an already created program
Save	Save current program
Save As...	Save current program under a new name
Save As Template	Save current program as a template
Save BAPS Sourcecode	Save QLL program under current name with extension '.qll'
Save BAPS Sourcecode As...	Save QLL program under a new name with extension '.qll'
Print...	Print program documentation
Print BAPS Sourcecode	Print content of the BAPS source text window
Printer Setup...	Configure used printer
Exit	Close graphical programming system
File list	List of the files last edited, see chapter 4.2.1.

Main menu item 'Edit'

Menu item	Signification
Undo	Undo last command
Redo	Undo last undo command
Cut	Cut marked area and transfer into clipboard
Copy	Copy marked area and transfer into clipboard
Paste	Insert content of the clipboard at cursor
Delete	Delete mark without transfer into the clipboard

Fundamental principles

Menu item	Signification
Find...	Search text within the source text program
Group	Group marked elements to one collective element
Ungroup	Represent elements regrouped in the collective element in the original state
Goto...	<p>Line Position cursor in the source text window on the selected line</p> <p>Next error Compile: After faulty compilation, position the cursor in the source text window on the faulty line</p> <p> debug operation: After faulty compilation, position the cursor in the source text window on the faulty line and mark the corresponding element in the PFC.</p> <p>Last Position Position the cursor in the source text window on the line last modified and mark the element in the PFC</p> <p>Show Actual Position Position in text operation the cursor in the source text window on the current line and mark the element in the PFC.</p>
Select all	Select all program elements of the currently visible level (except begin and end element)
More Options	<p>When selecting, a submenu with the following items appears:</p> <p>Insert Alternative Insert new alternative within CASE instruction</p> <p>Delete Alternative Delete alternative within CASE instruction</p> <p>Insert Parallel-Branch Insert new parallel branch within parallel instruction</p> <p>Delete Parallel-Branch Delete parallel branch</p> <p>Insert New Command Open dialog to insert a user-defined command (Macro)</p>

Fundamental principles

Main menu item 'Item'

Menu item	Signification
Parameter...	Open parameter dialog for the current element at the cursor
Properties...	Open properties dialog for the current element at the cursor
Forward	Move cursor to the next element
Backward	Move cursor to the previous element
Left	Move cursor to the left
Right	Move cursor to the right
Level Down	Activate next lower program level in the PFC
Level Up	Activate next upper program level in the PFC
Color...	Call dialog for the color selection

Main menu item 'View'

Menu item	Signification
Arrange Layer	Re-arrange elements of the current program level
Arrange All	Re-arrange elements in all program levels
View BAPS Source code	Show or hide BAPS source text window
View Hierarchy	Show or hide level display
Zoomwindow	Show or hide zoom window
View All Icons ...	Show and rearrange existing icons in one window, see page 4-18
Full Screen	Switch display in the full screen mode

Fundamental principles

Main menu item 'Debug'

Menu item	Signification
Select Program	Activate test mode for current program
Start/Continue Program	Start selected program on control or resume program execution after reaching a break point
Stop Debugging	End test mode
Start Debug-Mode	Activate test mode without selecting current program
Single Step	Select individual program command and branch in subroutine if the program is a subroutine
Show Actual Position	In debug operation, position the cursor in the source text window on the current line and mark the corresponding element in the PFC
Toggle Breakpoint	Set or delete the break point at the current position in the PFC
Remove All Breakpoints	Delete all break points
Watch Signals...	Select signals for display in the observation window
Watch Variables...	Select variables for display in the observation window
Watch External Process variables...	Select variable from external processes for display in the observation window

Main menu item 'Library'

Menu item	Signification
Administrate...	Select loaded libraries
Edit icon arrangement...	Change position of an icon within the icon bar Create new icon or new icon bar
Export...	Create export library
Import...	Import export library
Reload...	Re-load under 'Library – Administrate...' selected libraries

Fundamental principles

Main menu item 'Coupling'

Menu item	Signification
Copy rho -> PC...	Load files from control
Copy PC -> rho...	Copy files in control

Main menu item 'Extras'

Menu item	Signification
Examine Parameters	Search for incompletely parameterised symbols
Start DIM...	Start Data Input Manager
Compile...	Compile current source program
Edit Tools...	Enter new tools or edit existing ones
Save Tools...	Save existing tools in a file
Edit Commands...	Enter new commands or edit existing ones

Fundamental principles

Main menu item 'Options'

Menu item	Signification
Check BAPS Sourcecode	Switch on or off the automatic program test in the background
Include Comments	Adopt comments from the PFC into the BAPS source code
Mark BAPS Sourcecodes	Mark in the source text window the lines belonging to the PFC element
Show Linenumbers	Display current line number in the source text window and in the status bar
Follow Runposition	During the program run display the positions in the sublevels
Autosave...	Configure Autosave adjustments
Directories...	Adapt directories
Quickinfo	Switch on or off quick info window
Iconbar	Adapt icon bar operation mode. Following subpoints are available: Learning Switch on or off learn mode Reset Reset reference counter
Rho-Setup...	Configure connection to the control
Fonts	Change used font. When selecting the menu item, a submenu with the following items appears: Inside Elements... Define font used in the PFC symbols Comment... Define font used in the PFC comments BAPS Sourcecode ... Define fonts used in the source text window
Colors	Change used colors. When selecting the menu item, a submenu with the following items appears: Caret Change color for the cursor Breakpoints Change color for the break points Runposition... Change color for the execution position

Fundamental principles

Main menu item 'Tools'

Example since the tool menu can be freely configured by the user. For detailed information see section 'Main menu/Tools'

Menu item	Signification
Editor	Call ASCII-Editor
Online	Call Module 'Online'
Commandline	Open MS-DOS window

Main menu item '?'

Menu item	Signification
Manual BAP-Splus	Display BAPSplus manual in file format *.pdf 📄 Acrobat Reader must be installed
Tips and Tricks	Display tips for working with BAPSplus
Brief Description (BAPS)	Display manual BAPS3 short description in file format *.pdf 📄 Acrobat Reader must be installed
Programming Guide (BAPS)	Display manual BAPS3 Programming manual in file format *.pdf 📄 Acrobat Reader must be installed
Info...	Display program information, version number and copyright

The tool bar

The tool bar permits a direct access to the different menu items with their functions (e. g. New file, Load, Save...).



The following functions are made available (from left to right):

- create new file
- open file
- save file
- print file
- select next element in the PFC

Fundamental principles

- select previous element in the PFC
- select sub-level
- select next higher level
- delete level
- group elements
- define kinematic data (only possible in main program)
- enter program parameters (only possible in main program)
- define constants
- define types
- define variables
- define signals
- define subroutines (only possible in main program)
- display processing list
- define tool data
- edit commands
- compile current file
- start/continue program
- set/ delete interruption
- display signals
- display variables
- display external variables
- exit test
- List of the declared functions (opened in the screen)

Icon bar

The icon bar (or symbol bar) contains symbols (icons) – ordered by groups – which can be inserted into the program flowchart as often as desired by clicking with the mouse.

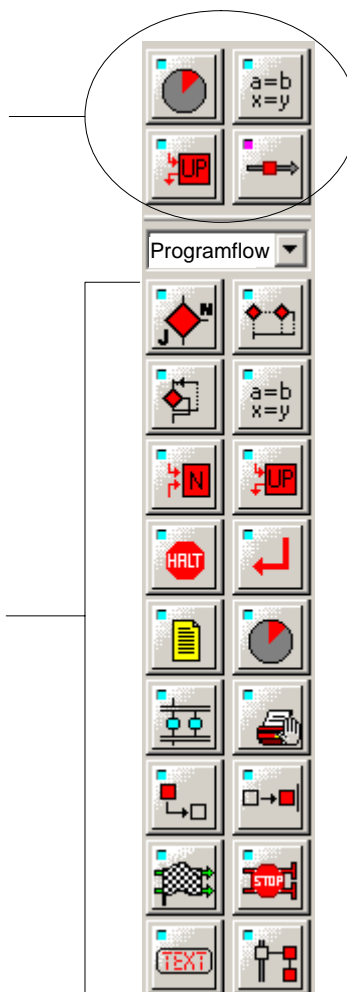
The icon bar consists of max. 20 columns, but only one at a time can be selected directly. This visible column is selected with the Combo box at the top rim, but can also be changed through the mouse menu (clicking with the right-hand mouse key) or using the scroll bar at the bottom rim.

Fundamental principles

Combo box to change the visible icon bar

Field of the four most frequently used icons, see section 5.6.

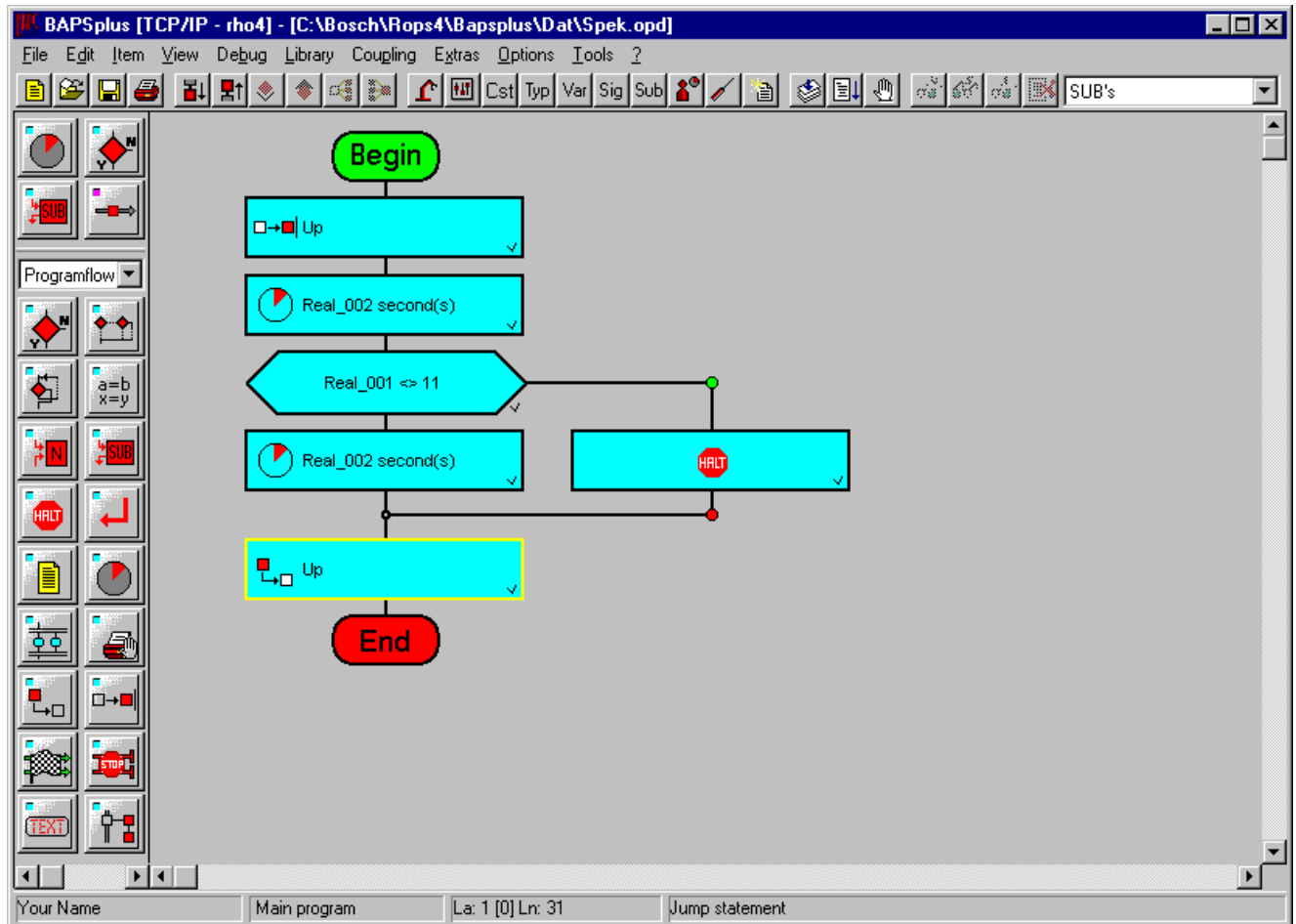
Current icon bar, in this case 'Programflow'



Fundamental principles

The programming section

The programming section represents the working section in which the program flowchart (short PFC) is created. It requires the major part of the main window.



Fundamental principles

The status line



The current information is displayed in the status line at the bottom rim of the main window.

- Name of current user
- Selected program level
- Numbers of the program and sub-levels and the current line number
- Explanation field with the following information – depending on the situation:
 - Explanation of the active symbol in the program flowchart
 - Error message of the integrated compiler
 - Status messages

A double click on the different areas in the status line enables a rapid access on the following functions:

User:	opens the user dialog
Level designation:	jumps to the BEGIN element of the program or subroutine
Level / line display:	opens the dialog 'Go to ...'
General info area:	shows the parameter dialog of the current element

3.2.1 Optional window

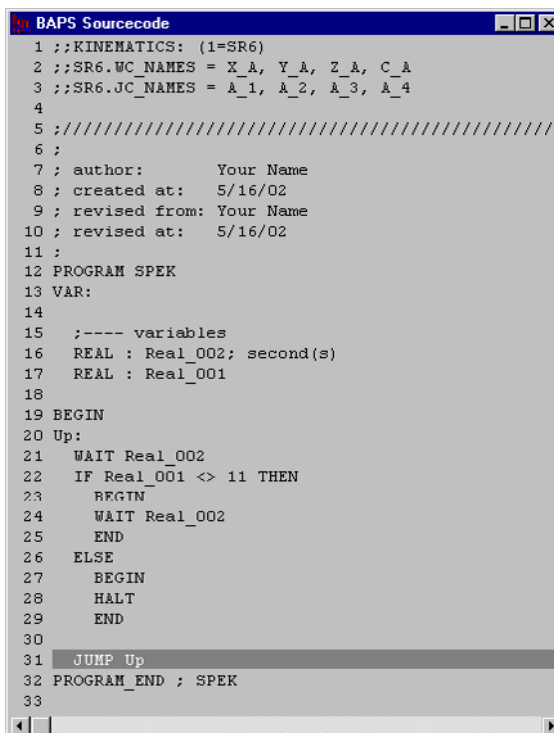
Additionally to the presentation of the graphical PFC, further windows can be opened by means of 'Menu/View'. These optional windows 'remember' their last position and size so that when opening them the next time, they will be displayed in the same size and at the same place.

BAPS source text

In the BAPS source text window, the automatic creation of the corresponding BAPS program text – the QLL program – can be followed simultaneously.

When the menu point 'View – Mark BAPS Sourcelines' is active, the source line of the current element is inverted in color. Lines numbers can be displayed in addition since BAPSplus can also work without them.

Fundamental principles



```

1 ;;KINEMATICS: (1=SR6)
2 ;;SR6.WC_NAMES = X_A, Y_A, Z_A, C_A
3 ;;SR6.JC_NAMES = A_1, A_2, A_3, A_4
4
5 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
6 ;
7 ; author:      Your Name
8 ; created at:  5/16/02
9 ; revised from: Your Name
10 ; revised at:  5/16/02
11 ;
12 PROGRAM SPEK
13 VAR:
14
15 ;---- variables
16 REAL : Real_002; second(s)
17 REAL : Real_001
18
19 BEGIN
20 Up:
21   WAIT Real_002
22   IF Real_001 <> 11 THEN
23     BEGIN
24       WAIT Real_002
25     END
26   ELSE
27     BEGIN
28       HALT
29     END
30
31   JUMP Up
32 PROGRAM_END ; SPEK
33

```

☞ If another line is selected in the source text window, the corresponding symbol in the PFC is activated. If this element is not visible, the program flowchart is scrolled accordingly.

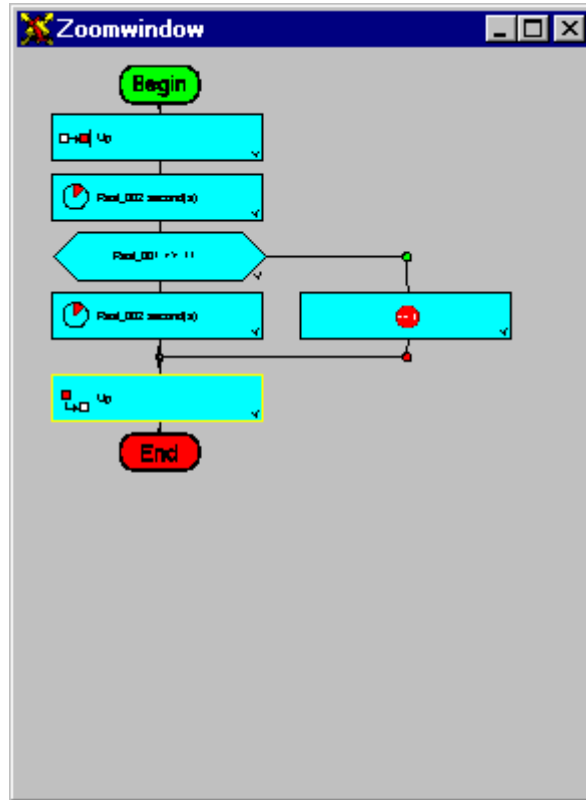
☞ In order to use the source text window e.g. only the consideration of the declaration part, the function 'Mark BAPS Sourcelines' in the menu 'Options' can be switched off. The other possibilities of the source text window are kept.

Zoom window

The overview display, (zoom window), shows the current program level (main program, subroutine or sub-level) completely.

While the programming section of the program flowchart can only show one screen page at a time, the complete level is displayed here in reduced scale, giving thus a quick overall view of the structure of the current level.

Fundamental principles

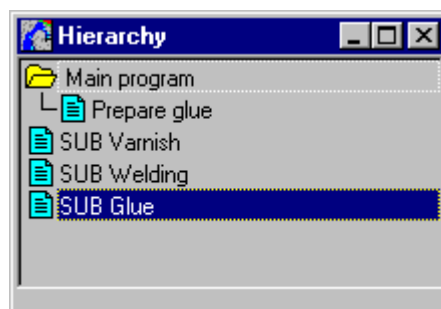


- ☞ The size of the zoom window can be changed to achieve an ideal representation or utilization of the screen section being available. In this window it is also possible to change the current symbol.

Hierarchy

The level display serves the hierarchic representation of all levels of the current program. Similar to the file tree in the Windows File Manager or Explorer, the levels can be displayed or hidden.

After a double click on the element, the level is displayed in the PFC and can be edited.



Fundamental principles

- ✎ **The size of the level display can be adapted to the individual requirements to ensure an ideal presentation or utilization of the screen section. In this window, too, it is possible to change the current symbol.**

Fundamental principles

Notes:

Elementary functions

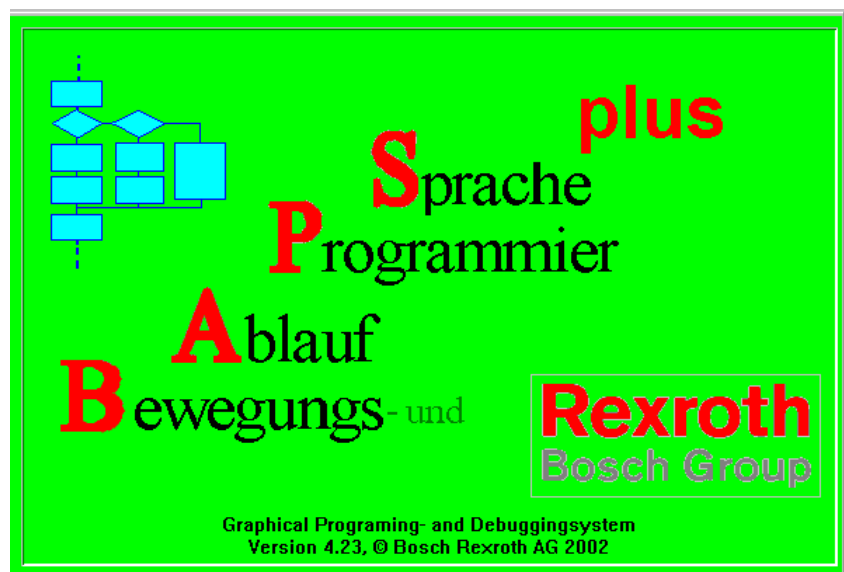
4 Elementary functions

This chapter describes in detail all BAPSplus functions including all menu items in the menu.

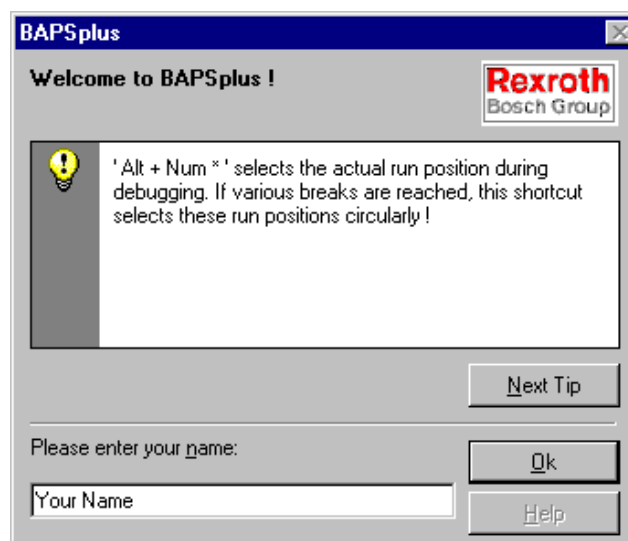
4.1 Start BAPSplus

- ★ Select the menu entry 'Create/BAPSplus' in **ROPS4** to start BAPSplus.

While the application is loaded a start dialog appears which is automatically closed after the successful start.



Then the welcome dialog appears.



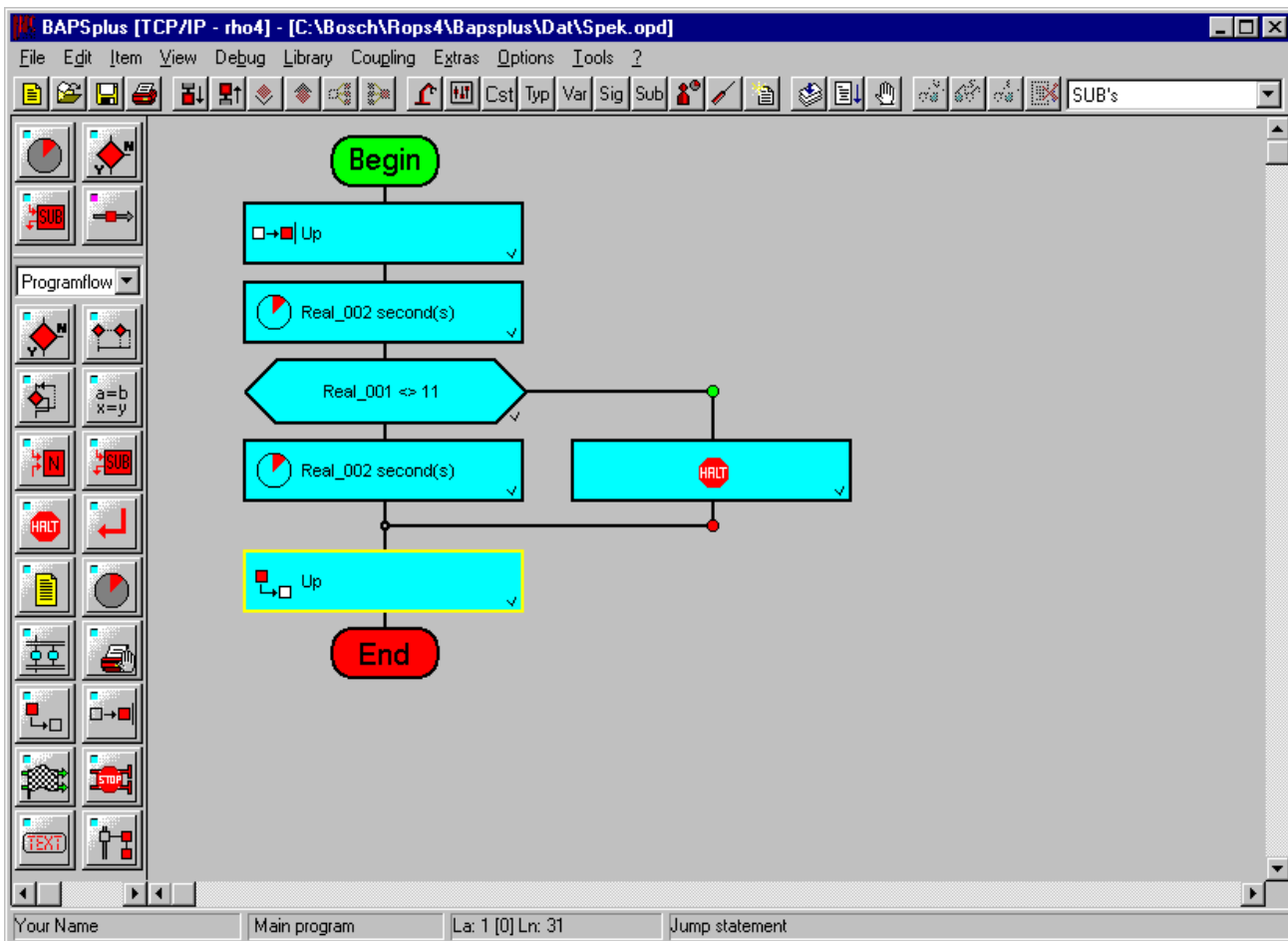
Elementary functions

- ★ Enter the user name to end the dialog.

The user name serves the logging of the created changes, see section 5.1.8.

- ☞ **At the next program start, the last entered name is adopted. It can be directly taken over with 'OK'.**
- ☞ **Press button 'Next tip' to get useful information.**

After a successful first program start, the BAPSplus main window appears with a new file on the screen (file name 'Noname.opd'). At every other start, the last edited file is directly loaded (in the screen: 'Spek.opd').



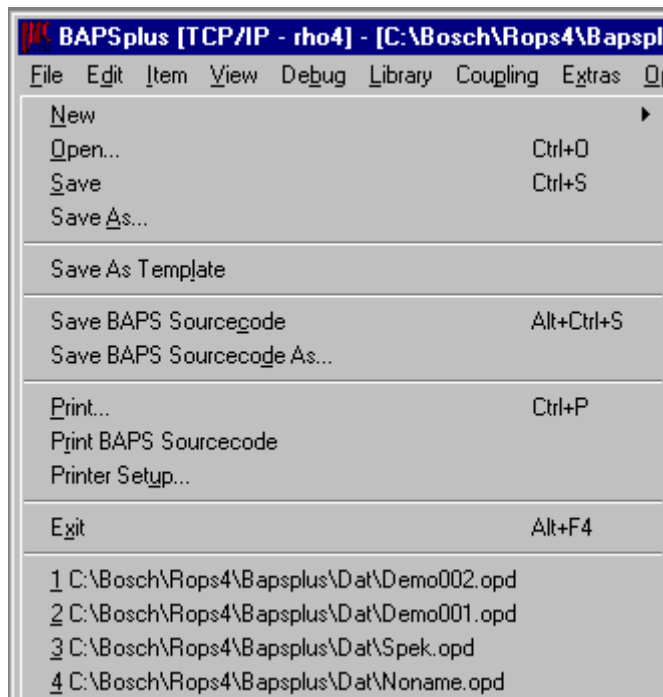
Elementary functions

4.2 Main menu items

4.2.1 Main menu item 'File'

Under this menu item, files can be created, opened, saved, printed and BAPSplus can be terminated.

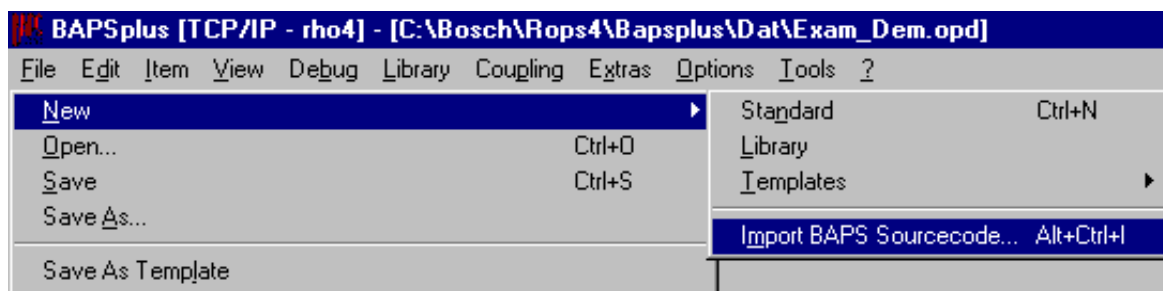
By selecting 'File' the following pull-down menu appears:



☞ **At the bottom of the file menu, a list of the last edited files appears. The associated files can be quickly opened by entering the corresponding number.**

New

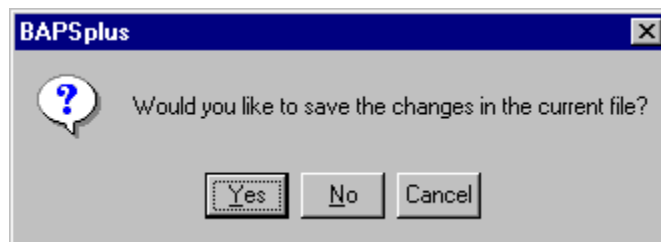
'New' enables to create a new file. After selection, the following submenu appears:




Elementary functions

Standard	Creation of a new program file with the current standard defaults from the files DEFAULT.owd and DEFAULT.okd
Library	Creation of a new library, see section 5.5.
Templates	Creation of a new program file based on a template from the template directory, see section 5.4.
Import BAPS Sourcecode...	Adopt source text from BAPS-QLL program that already exists, see chapter 7.

If the current file has been modified before a new file is to be created, the following message appears:



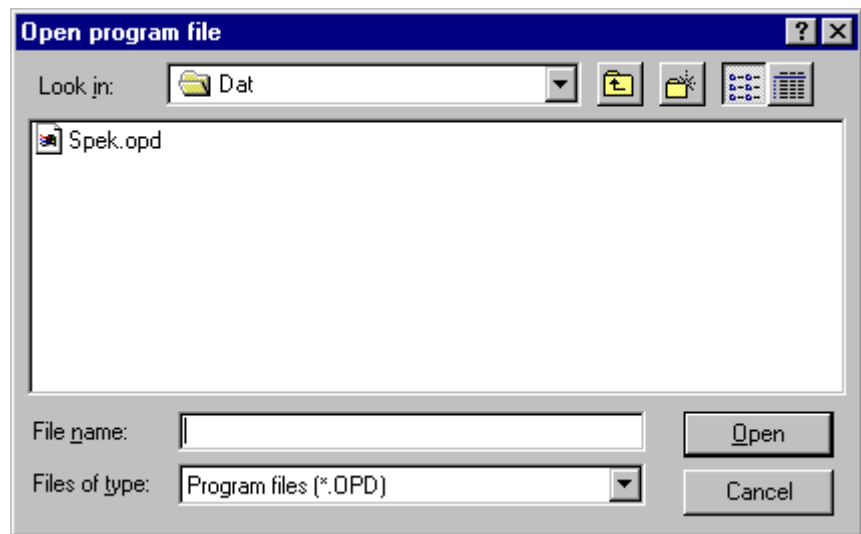
- ★ Click on 'Yes' to save the changes, 'No' to cancel the changes. With 'Cancel', the current process (i.e. creation of a new file) is aborted and the changed file remains open.
- ☞ **With the keyboard shortcut <Ctrl+N>, a new file, based on the default settings can be directly created.**
Associated symbol in the tool bar: 

Open...

This menu item opens the selected file or library. If the file is not existing or is not located in the indicated directory, a corresponding message appears. 'Create new file' is not possible here. The file is directly imported if it is a QLL-File (valid licence required).

Elementary functions

After selection of 'Open...' the following dialog appears:



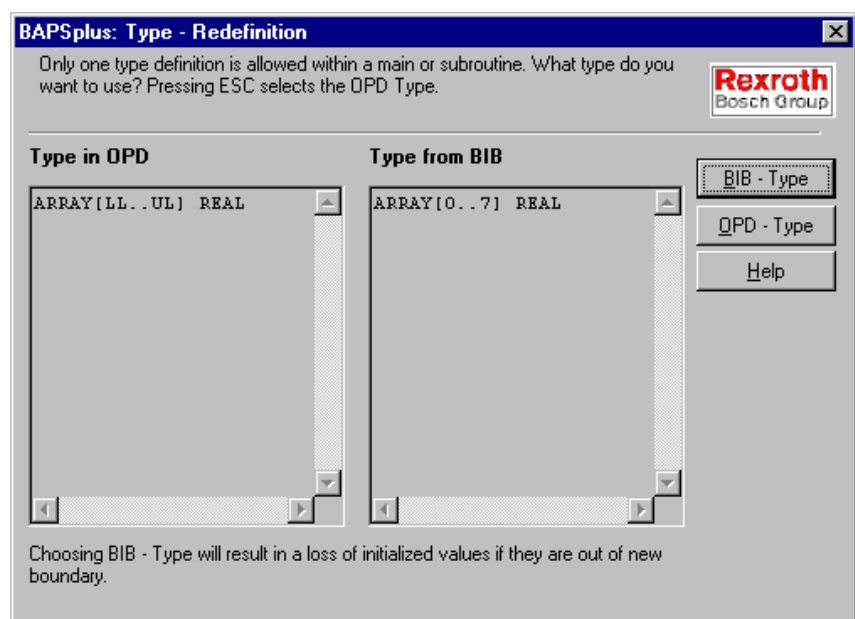
- ☞ **<Ctrl+O> opens directly the dialog.**
Associated symbol in the tool bar:



- ☞ **One of the files last used can be opened by clicking on its name down in the menu 'File' or by entering the corresponding number.**

When loading, all library types are also checked for changes. If there are modifications, the next dialog appears enabling to use the changed type from the library or work further with the previous one saved in the program.

Example:
Definition change of a field



Elementary functions

Save

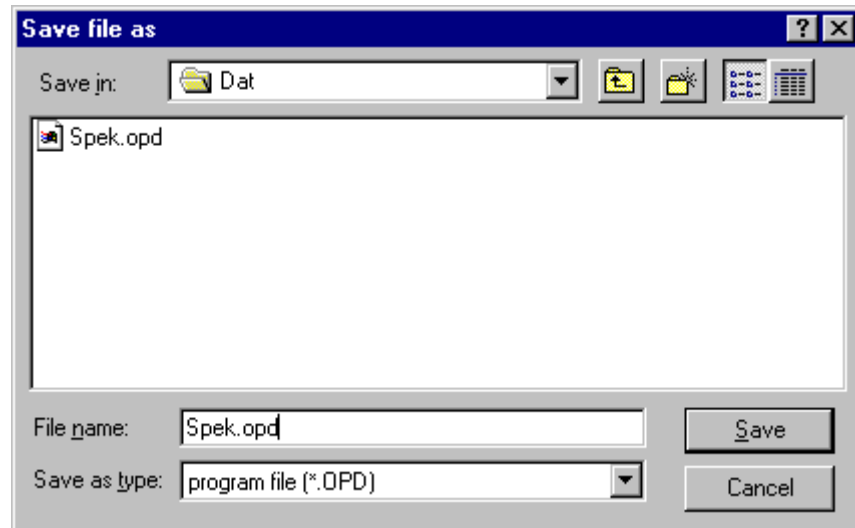
This menu item saves the active file or library. If the file or library is still without name, BAPSplus displays the dialog field 'File/Save as...', see 'Save as...'.

☞ **<Ctrl+S> stores the file directly.**
Associated symbol in the tool bar:

**Save as...**

'Save as...' saves the current file or library under a new name. A new name as well as a new destination directory can be selected.

The path indicated in the entry field 'Program files' under 'Options / Directories...' is preset as the directory.

**Save As Template**

Saves the current program in the BAPSplus template directory. This file is available as template under 'File/New/Templates'.

☞ **The saving is performed with the current name and without further inquiry messages. Existing files may be overwritten.**

Save BAPS Sourcecode

Saves the BAPS sourcecode belonging to the current program in a file in the code directory. The file name is composed of the name of the current file and the file extension '.qll'.

If the active program has not been saved, the dialog field 'Save BAPS Sourcecode As' (menu 'File') appears, see page 4-7.

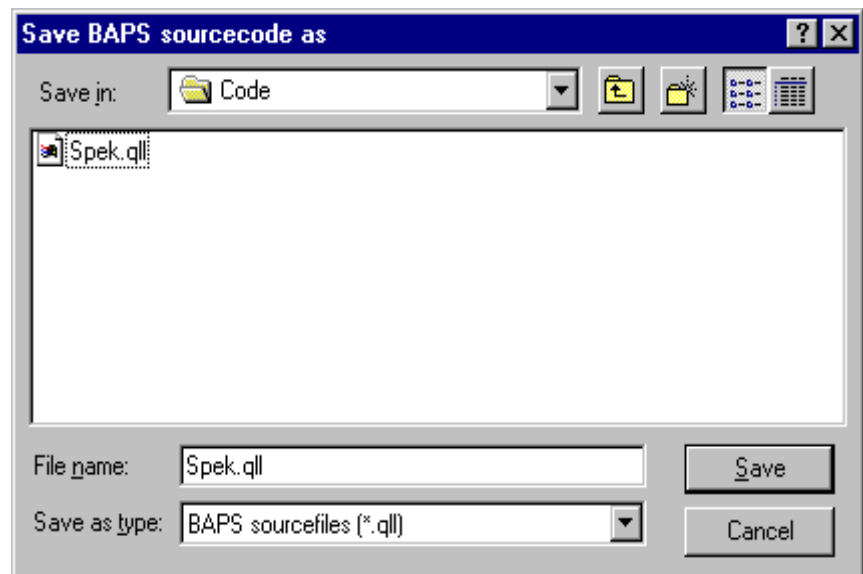
Elementary functions

The source code is saved in a way as it can be seen in the window. This means that comments are only saved if they can be seen in the window.

☞ **The program code is directly saved with <Alt+Ctrl+S>.**

Save BAPS Sourcecode As ...

Saves the BAPS source text under a new name. A new name as well as a new target directory can be selected.

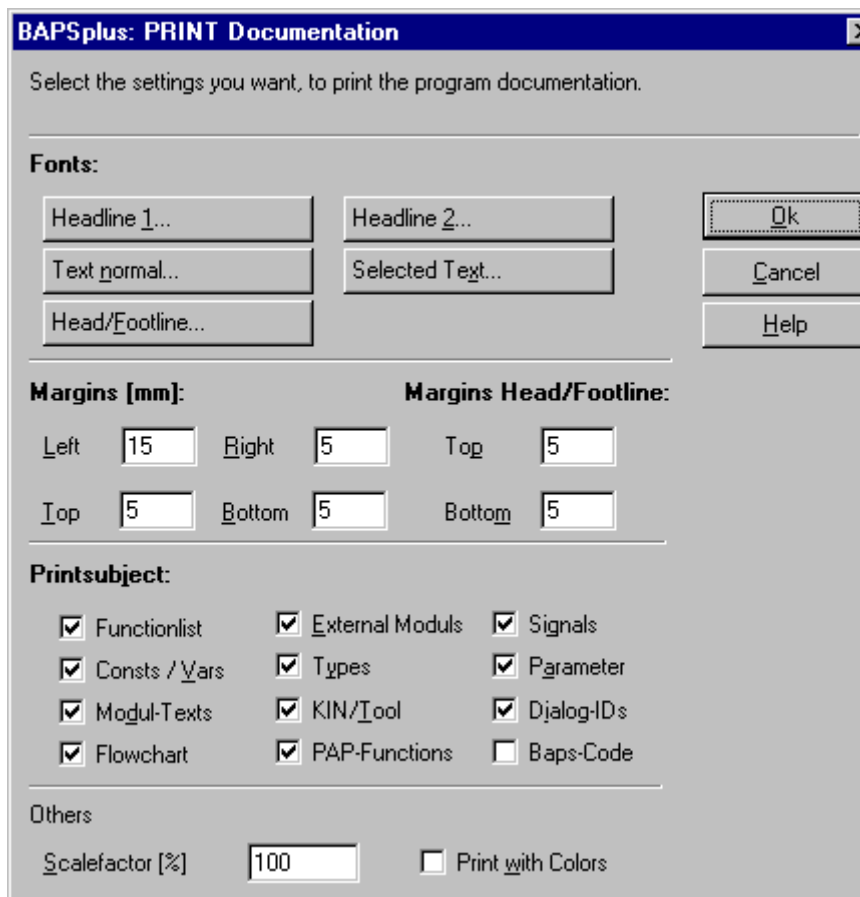


Elementary functions

Print...

'Print...' prints the program documentation. In the corresponding dialog, it is possible to select which program parts are to be printed in which way.

 **<Ctrl+P>** starts directly the print dialog.



BAPSplus: PRINT Documentation

Select the settings you want, to print the program documentation.

Fonts:

Headline 1... Headline 2... **Ok**

Text normal... Selected Text... **Cancel**

Head/Footline... **Help**

Margins [mm]: **Margins Head/Footline:**

Left 15 Right 5 Top 5

Top 5 Bottom 5 Bottom 5

Printsubject:

Functionlist External Moduls Signals

Consts / Vars Types Parameter

Modul-Texts KIN/Tool Djalog-IDs

Flowchart PAP-Functions Baps-Code

Others

Scalefactor [%] 100 Print with Colors

The following settings are available:

- Fonts for the headlines, for the normal text, the highlighted text and the header and footline
- Margins
- Header and footline distance
- Scale factor
- Print subject. Available for selection are:
 - List of the subroutines contained
 - List of the used external programs
 - List of the used constants and variables
 - List of the used signals
 - List of the used types
 - Program parameters
 - Program comment or module texts

Elementary functions

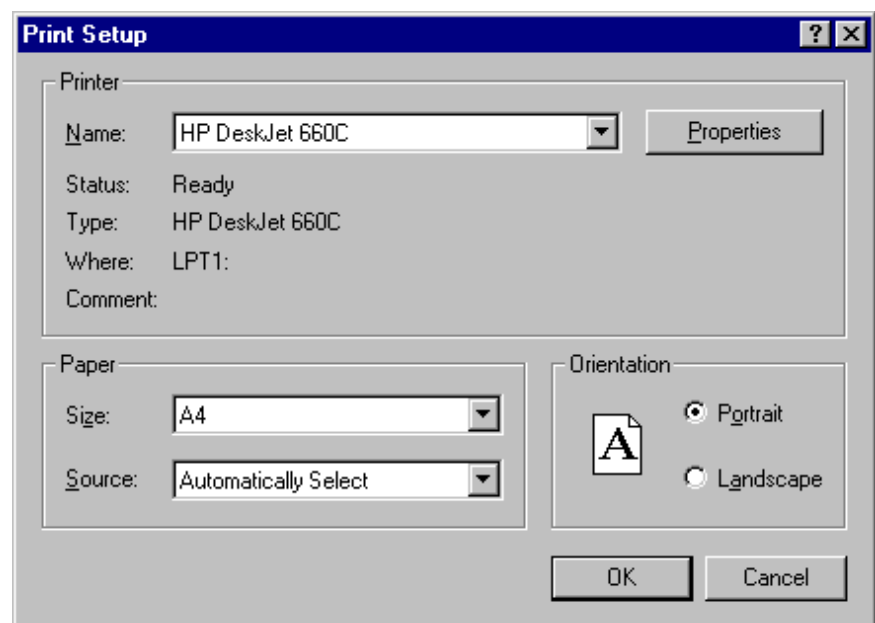
- Used kinematics and tools
- ID of the automatically defined dialogs
- Program flowcharts
- PFC functions
- BAPS source texts (QLL program)

Print BAPS Sourcecode

Prints the content of the BAPS source text window with number of line, if it is preset.

Printer Setup...

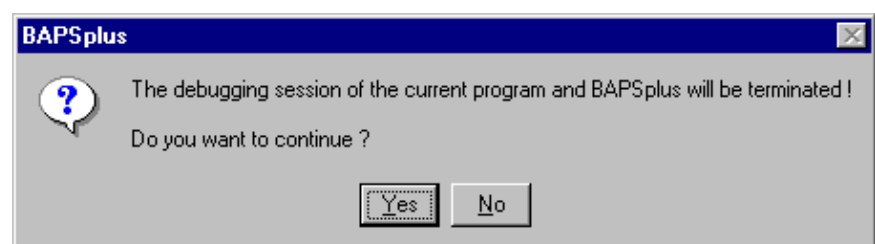
'Printer Setup...' enables to configure the printer to print the program documentation.



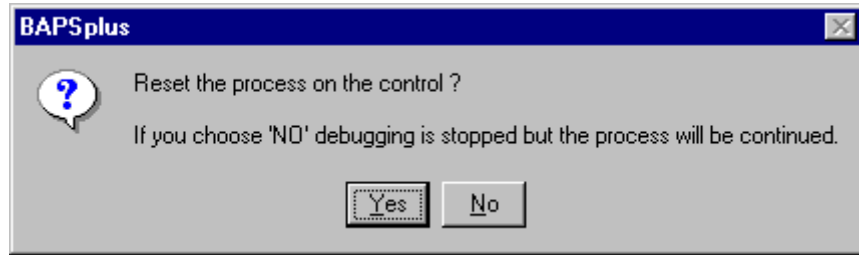
Exit

'File/Exit' closes BAPSplus. All opened windows are closed.

If the text mode is active, the following dialog appears giving the possibility of closing the program or cancelling the shutdown of the programming system.



Elementary functions



 **If the current file has been changed and not saved yet, a corresponding message appears, see page 4-4.**

4.2.2 Main menu item 'Edit'

'Edit' allows to arrange commands for the modification of the PFC. If the function is available, depends on the currently selected symbol.

By selecting 'Edit', the following submenu appears:



Undo

Undoes the last command. The number of the UNDO steps is limited through an entry in the file 'BAPSplus.ini'. It means when '10' is entered there, only the last 10 changes can be undone.

 **Shortcut: <Ctrl+Z>.**

Elementary functions

Redo

Carries out again the last undone command.

 **Shortcut: <Ctrl+Y>.**

Cut

Cuts the marked area and transfers it into the clipboard. The deposit format is an internal BAPSplus format. This means that the content of the clipboard can only be used by BAPSplus.

 **Shortcut: <Ctrl+X>.**

Copy

Copies the marked area and transfers it into the clipboard. Contrary to the command 'Cut', the marked elements are not removed.

 **Shortcut: <Ctrl+C>.**

Paste

Inserts the content of the clipboard at the place of the cursor. This command is only available when symbols from the PFC are contained in the clipboard.

 **Shortcut: <Ctrl+V>.**

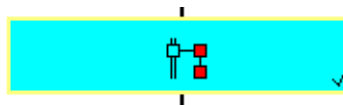
Delete

Delete the marked area without transfer into the clipboard.

 **Shortcut: .**

Group

The marked elements are assembled in a collection element and grouped in a sublevel. It is symbolized in the PFC by a block element.



Elementary functions

Editing this sublevel is possible with double click or – if this block element is the active element – with <Enter>.

 **Shortcut: <Ctrl+G>.**

Ungroup

Represents the elements assembled in a block element under 'Group' in their original state again.

 **Shortcut: <Ctrl+U>.**

Goto

When selected, a submenu containing the following items appears:

Line

Positions the cursor in the source text window on the selected line.

 **Shortcut: <Ctrl+L>.**

Next error

Compile:

Selects after a faulty compilation the erroneous element and shows the corresponding line in the source text window.

Debug mode:

Selects the corresponding element in case of a process error and shows the corresponding line in the source text window.

 **Shortcut: <Ctrl+Shift+N>.**

Last edited place


Positions the cursor in the source text window on the last modified line and marks in the PFC the corresponding element.

 **Shortcut: <Shift+F5>.**

Elementary functions

Display execution position

Positions in the debug mode the cursor in the source text window on the current line and marks in the PFC the corresponding element.

 **Shortcut: <Alt+*> in numerical keypad.**

Select All

Marks all program elements except initial and final elements.

 **Shortcut: <Ctrl+A>.**

More Options

When selected, a submenu containing the following items appears:

Insert alternative

Inserts a new alternative within a CASE instruction.

 **Shortcut: <Alt+Ins>.**

Delete alternative

Deletes an alternative within a CASE instruction.

 **Shortcut: <Alt+Del>.**

Insert parallel branch:

Inserts a new parallel branch within a parallel instruction.

Delete parallel branch:

Deletes a parallel branch.

Insert new command:

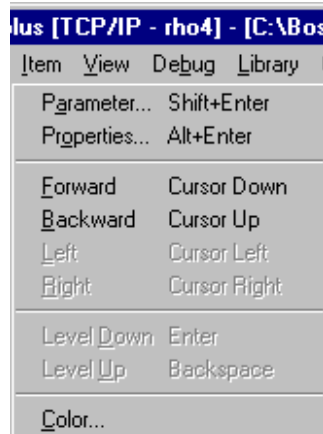
Opens dialog to insert a user-defined command.

Elementary functions

4.2.3 Main menu item 'Item'


The functions for modifying the active symbol in the PFC are grouped under this menu item. Whether the functions are available, depends on the currently selected symbol.

When selecting 'Item', the following submenu appears:

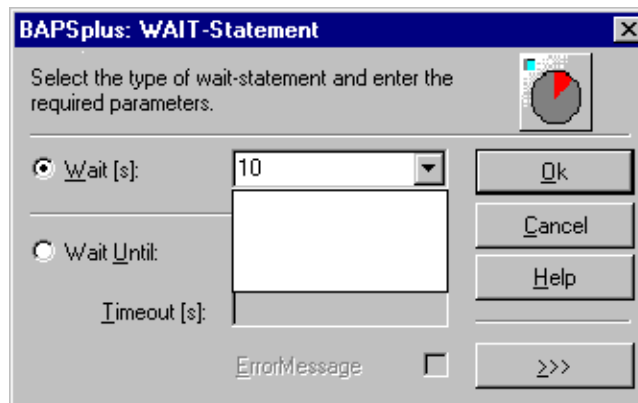


Parameter...

'Parameter...' opens the parameter dialog for the current element at the cursor. In this dialog, the data belonging to the current command can be entered.

 **Shortcut: <Shift+Enter>.**

Example: Parameter dialog (WAIT)

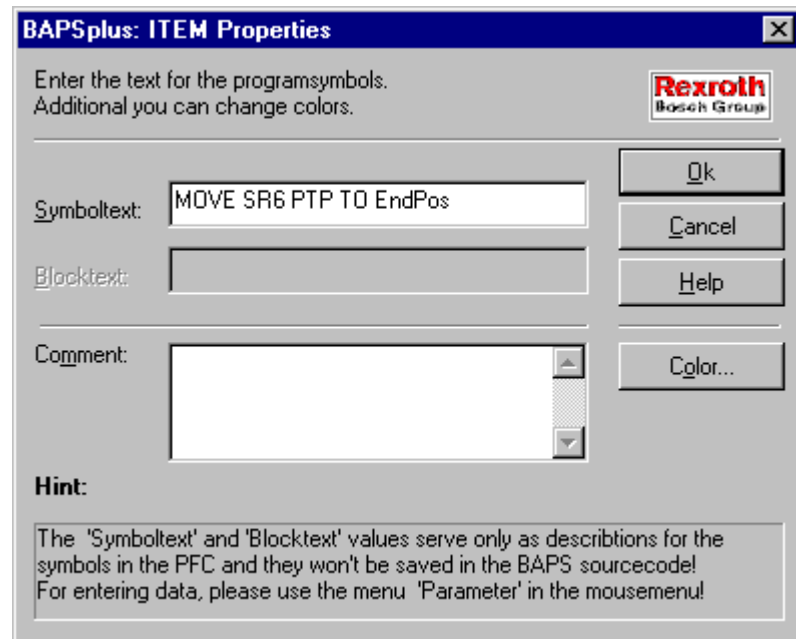


Properties...

'Properties...' opens the property dialog for the current element at the cursor.

 **Shortcut: <Alt+Enter>.**

Elementary functions



Whether entry fields can be modified depends on the corresponding element. Text fields that cannot be edited are shown as 'pressed'.

- ☞ **The symbol text is adapted through the parameter dialog to current settings if it is not deactivated in the dialog.**

Forward

Moves the cursor to the next element.

- ☞ **Other possibility: key <Arrow down>.**

Backward

Moves the cursor to the previous element.

- ☞ **Other possibility: key <Arrow up>.**

Left

Moves the cursor to the left.

- ☞ **Other possibility: key <Arrow left>.**

Elementary functions

Right

Moves the cursor to the right.

 **Other possibility: key <Arrow right>.**

Level Down

Activates the next lower program level in the program flowchart. This command is only available if a block element is activated.

 **Other possibility: key <Enter>.**

Level Up

Activates the next higher program level in the PFC. Can only be activated if a sublevel is represented in the PFC.

 **Other possibility: key <Backspace>.**

Color...

Calls the dialog for colour selection. With this dialog the background colour of the elements in the program flowchart can be individually selected. In basic setting, the elements are shown with a light blue background.

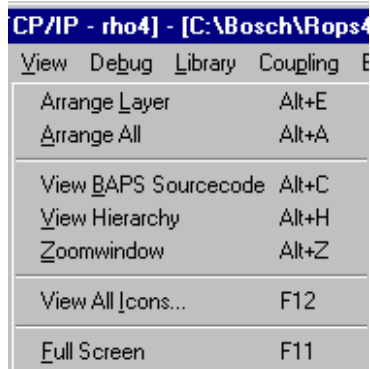


Elementary functions

4.2.4 Main menu item 'View'

'View' delivers functions for activating the possible types of representation and arranging manually the program flowchart.

When selecting this menu item, the following submenu appears:



Arrange layer

Through Drag & Drop with the mouse, it is possible to arrange individually the symbols in the PFC and re-arranges the elements of the current program level.

 **Shortcut: <Alt+E>.**

Arrange All

Re-arranges all elements in all program levels, regardless of whether the level in the PFC can be seen or not.

 **Shortcut: <Alt+A>.**

View BAPS Sourcecode

Shows the BAPS source text window and hides it again. In this window, the automatic code text creation can be simultaneously monitored.

 **Shortcut: <Alt+C>.**

View Hierarchy

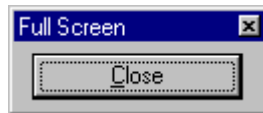
Shows the view hierarchy and hides it again, see page 3–22.

 **Shortcut: <Alt+H>.**

Elementary functions

Full Screen

Switches the display in full screen mode. In this mode, only the program flow chart is represented, the icon bar, the menu etc. cannot be seen. The key "ESC" or a mouse click on the button "Close" closes the full screen mode.



 **Shortcut: <F11>.**

4.2.5 Main menu item 'Debug'

The comprehensive test possibilities of BAPSplus are grouped under this menu item. The execution of the program can be influenced, signals and variables observed (monitored) and breakpoints set.

When selecting the menu item, the following submenu appears:



Select Program...

This menu item activates the debug mode for the current program. If the program has been changed or does not yet exist, the inquiry appears whether the file is to be translated and transferred to the control. Then the program is selected on the control, i.e. it is prepared for execution and the debug mode is activated. Now the remaining items of the debug menu are available.

Elementary functions


The program is then selected on the control (i.e. prepared for execution) and the debug mode is activated. The other items of the debug menu are available.

 **Shortcut: <F8>.**

Start/Continue Program

Starts the selected program on the control or continues the program execution after reaching a breakpoint.


After the selection of the program, it will not be automatically started on the control; this must be effected by selecting 'Start/Continue Program'. The program execution can be monitored by observing the movement of the cursor (colored frame around the current PFC symbol, or colored bar in the source text window).

 **Shortcut: <Shift+F8>.**
Corresponding symbol in the tool bar:



Stop debugging


Terminates the debug mode and closes all opened observation windows. An inquiry message appears asking whether the program on the control is also to be closed or not. In a further inquiry, the user must indicate if the break points on the control are to be deleted too.

 **Shortcut: <Ctrl+F8>.**
Corresponding symbol in the tool bar:



Start Debug-Mode

Activates the debug mode without selecting the current program. Only after the activation of the debug mode, break points are reported from the rho to BAPSplus.

 **The debug mode is deactivated via 'Stop Debugging'. The program execution can be monitored in this mode; this is only possible when the program is selected.**

Single Step


Allows the debugging of a program instruction after instruction. When selected, one program command is always executed at each time.

 **Shortcut: <F9>.**

Elementary functions

Show Actual Position


Determines if the current element is synchronized with the execution position. If the menu item is switched on, the current element (i.e. the cursor) is set on the execution position last reported.

 **Shortcut: <Alt+*> in numerical keypad.**

Toggle Breakpoint

Sets or cancels a breakpoint at the current position in the program flowchart. Breakpoints are characterized by a red frame with a broken line in the program flowchart. In the code text window they are set off by a dark bar.

If this command is carried out once more a breakpoint set before is deleted. Any number of breakpoints is admitted. The breakpoints are saved between the test runs; i.e. if a program is tested again, all available breakpoints are re-activated.

 **Shortcut: <Ctrl+B>.**
Corresponding symbol in the tool bar:

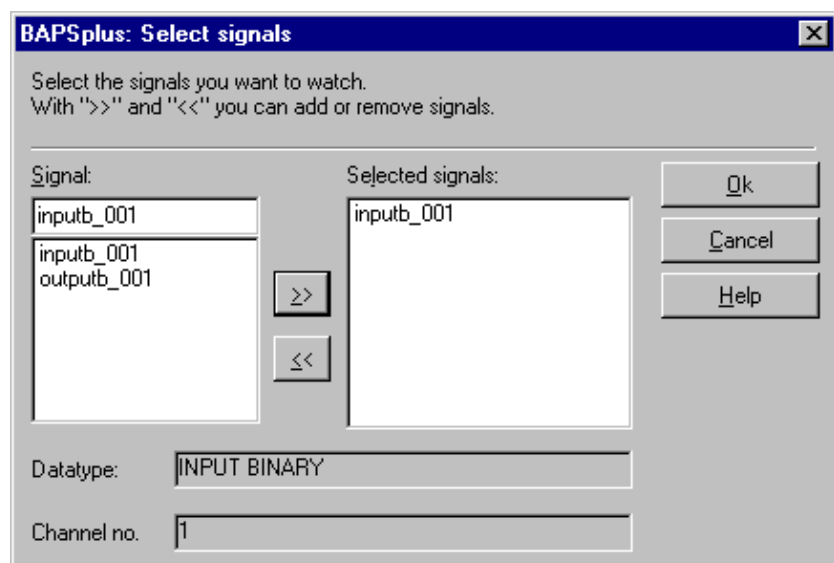


Remove All Breakpoints

Deletes all breakpoints set before in the current program.

Watch Signals...

Shows a dialog for selecting the signals which are to be displayed or watched in the observation window.



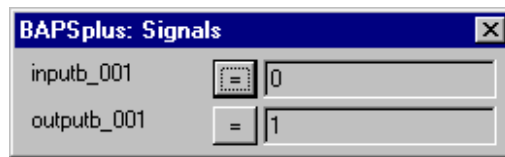
Elementary functions

In the list box 'Signals' all signals declared in the program appear. With '>>' the selected signal is transferred into the list box 'Selected signals'. With '<<' selected signals are removed.

☞ **Corresponding symbol in the tool bar:**



The selected signals are displayed in a separate observation window. The current value is in the edit fields (right). It is possible to enter new values. Modifications must be confirmed with '=' to adopt the new values into the program.



☞ **The DDE Server enables to watch up to 10 binary signals at the same time.**

Watch variables...

'Watch variables...' opens a dialog to select variables that are to be observed. This dialog has a similar structure as 'Watch signals'. It contains however in the left list box all variables declared in the program. the operation and the display of the selected variables are the same.

The maximum number of observed variables can be limited through its size since a 200-byte buffer is used to exchange variable values.

Example:

Should a text variable and five points of a 6-axis robot be displayed, exactly 200 points are used. No further variables can then be watched.

☞ **List of the memory requirement, see manual 'BAPS Programming manual', Function 'SIZE OF'.**

Record types cannot be displayed. The corresponding components must be listed:

RECORD.m_real1	:	shows 'm_real1' of the record variable 'Record'
P1.a_1	:	shows the axis information 'a_1' of the point P1
Sr6.P1.a_1	:	shows the axis information of a certain kinematic

Elementary functions

The procedure is similar for fields. It will be described by means of the same example as in the “Online-DDE-Server4 description”:

```
'ARRAY[1..30] ARRAY[1..10] INTEGER: INT_ARRAY'
```

Access to a field variable

```
'int_array[1][1]' ;Delivers an INTEGER value of the field 'int_array'
```

Access to complete field dimension

```
'int_array[1]' ;Delivers 10 INTEGER values of the field 'int_array'  
;'int_array'[1][1] to 'int_array'[1][10])
```

Access to area of a field dimension

```
'int_array[1][2-5]' ;Delivers 4 INTEGER values of the field 'int_array' ('int_array' [1][2]  
;'int_array'[1][3], 'int_array'[1][4] and 'int_array'[1][5])
```

or

```
'int_array[1-2]' ;Delivers 20 INTEGER values of the field 'int_array'
```

Not allowed is:

```
'int_array[1-5][2-5]' or 'int_array[1-5][2]'
```



Corresponding symbol in the tool bar:



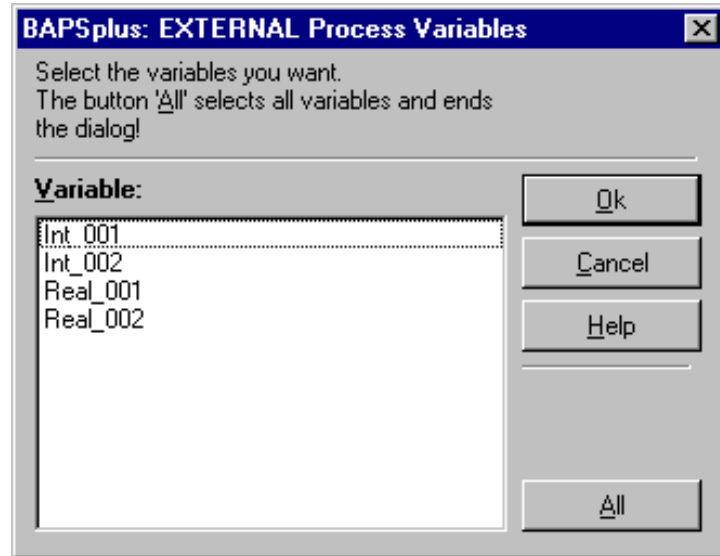
For further information, see manual 'DDE-SERVER 4'.

Watch External Processvariables...

BAPsplus allows to observe variables from any programs that are executed on the control. The desired program is selected via a dialog.

All variables of the desired program are then displayed in a selection dialog. If the selected program is not present on the control, a corresponding message appears instead of the observation window.

Elementary functions



- ★ Select the desired variable with mouse click. Another mouse click cancels the selection.

All variables are selected via 'All' and the selection dialog is directly closed.

The selected variables can then be watched in a separate window.

- ☞ **Corresponding symbol in the tool bar:**

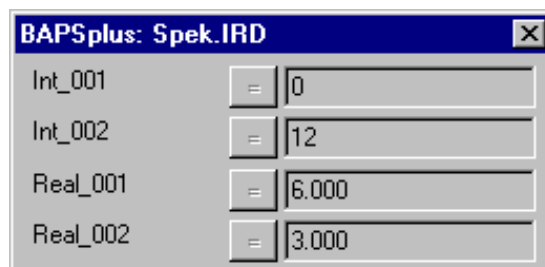


The program currently loaded can be loaded as an external one in order to have additional variables displayed in a separate window.

**CAUTION**

There is only one buffer for all variables.

Displaying variables of external processes can reduce the number of local variables.



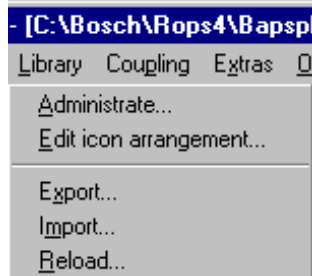
- ☞ **Variables of external processes cannot be modified.**

Elementary functions

4.2.6 Main menu item 'Library'


'Library' makes entries available allowing the selection and edition of the currently loaded libraries.

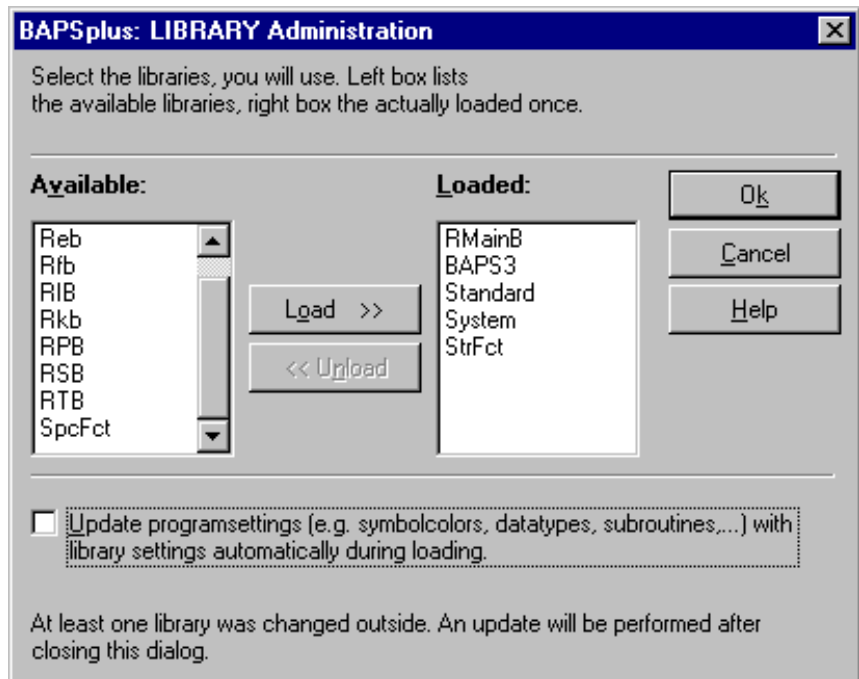
When selecting the menu item, the following submenu appears:



Administrate...

A dialog field with the available or currently loaded libraries appears. Libraries are loaded or removed via 'Load>>', '<<Unload' or a double-click on the name. All libraries saved in the LIB directory that are not loaded yet appear in the left list box (Available).

 **An automatic library monitoring of the changes is performed in the background. It is ensured in this way that even if libraries are replaced, it is worked with the latest state.**



Elementary functions

The loading of the libraries depends on the checkbox of the administration dialog.

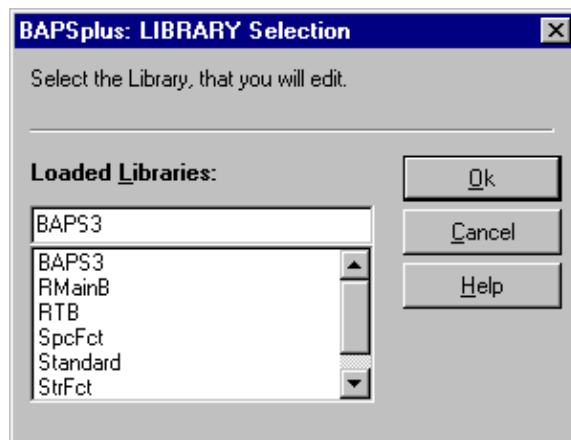
☞ **A maximum of 50 libraries can be integrated.**

☞ **A double click on the library name loads or unloads this library.**

Edit icon arrangement...

Enables the modification of the symbols contained in the icon bar.

After selection, a dialog listing all integrated libraries appears:



★ Select library via 'Ok' or double click.

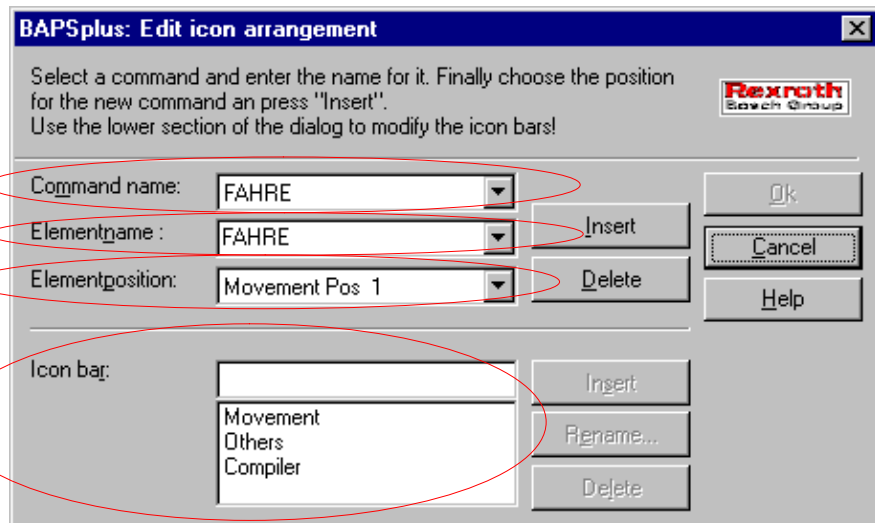
The actual edit dialog appears:

Selection of new functions


Selection of the element name

Free positions for selection

Combobox 'Icon bar'



Elementary functions

 **You can use your own icons for the commands. To this, the bitmap files must be copied into the user icon directory (see chapter 3.1.1 Directory structure). If the bitmap have to be displayed in the program flow chart, the name edited in “Elementname” (in the picture above “FAHRE.BMP) must be used. For use in the icon bar, the addition “Icon_” must be placed in front the bitmap filename (“Icon_FAHRE.BMP”)**

New commands from the selected library can be adopted in a desired icon bar. For each icon bar, two columns are available with 9 free positions each.

New icon bars can be created by editing a new name in the combo box 'Icon bar' and pressing the button 'Insert'. Press 'Delete' to remove it.

 **New symbol names and position within the icon bar can be chosen freely.**

- ★ End dialog with 'Close'.
If modifications have been made, an inquiry appears asking for saving.
- ★ Confirm with 'Yes' to save modifications, press 'No' to cancel the changes.

Export...

'Export...' generates an export library for the current program for the data exchange, see section 5.8.

After selection, a selection dialog for entering the library name appears. Default setting: name of the current file with the extension '.opx'.

 **Export libraries contain all commands and library functions used in the program and the arrangement of the symbols.**

Import...

'Import...' shows a dialog with libraries for loading. When a library is loaded, its commands are available, see section 5.8.

Reload...

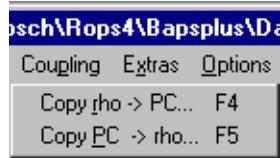
'Reload...' removes all export libraries loaded before and loads the original libraries integrated under 'Library/Administrate...'

Elementary functions

4.2.7 Main menu item 'Coupling'

'Coupling' contains functions to transfer files between PC and control.

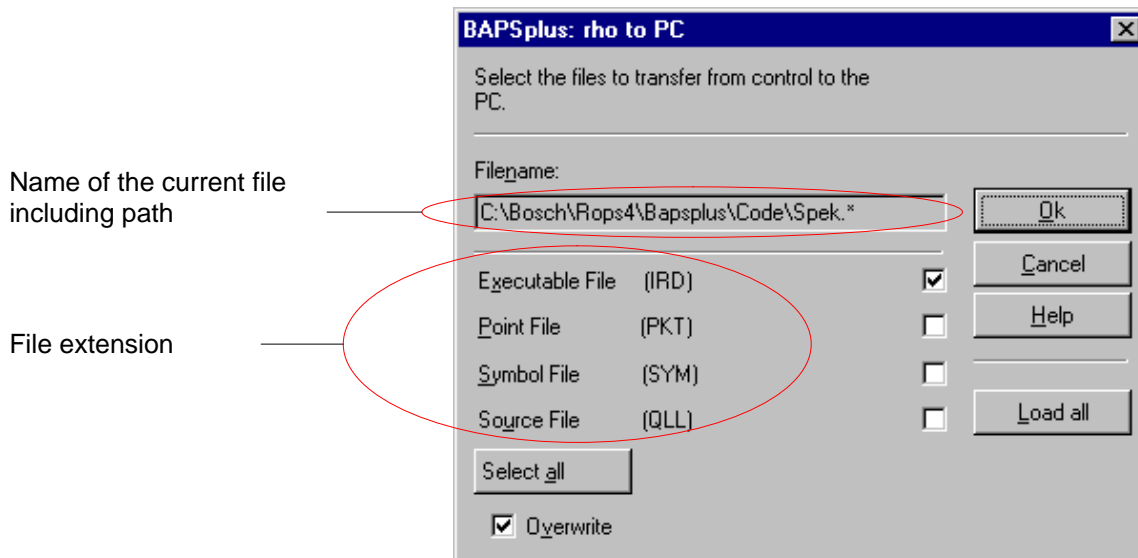
When selecting the menu item, the following submenu appears:



Copy rho -> PC...

'Copy rho -> PC...' starts to copy files from the control into the PC.

A dialog for selecting the files that are to be copied appears:



Name of the current file including path

File extension

- ★ Click on desired file extension.
All file extensions are selected via 'Select All'.
- ★ Start copying with 'Ok'.

BAPSplus checks

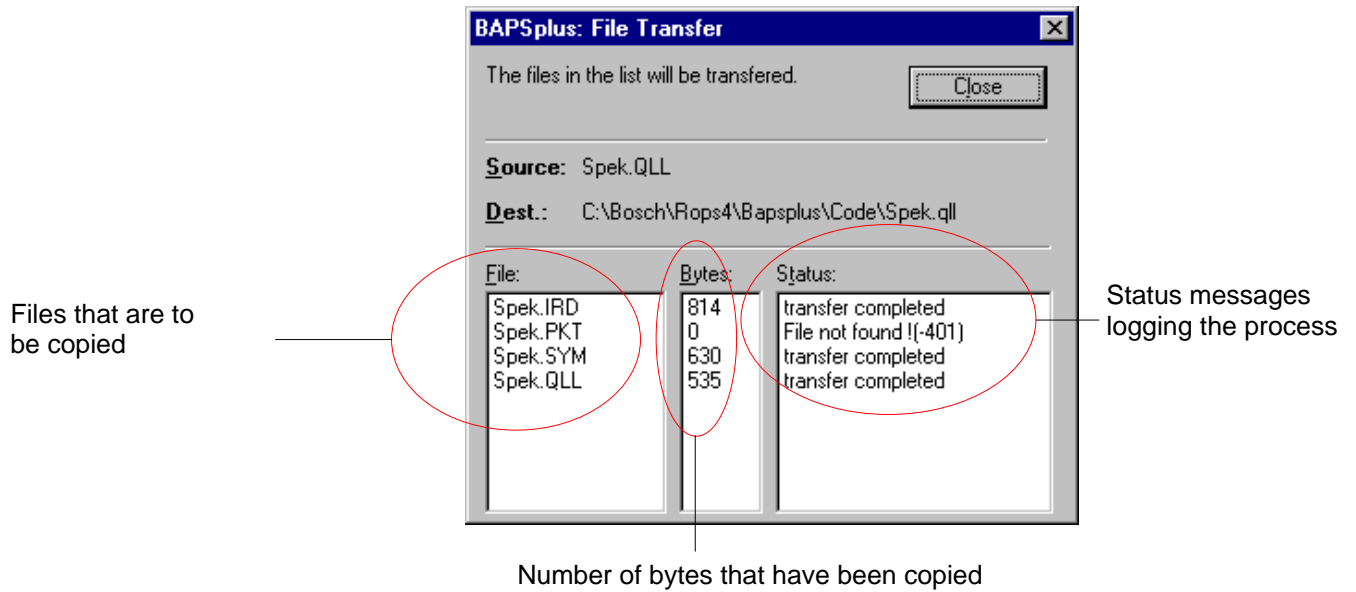
1. if the desired file exists on the control. If not, a corresponding message appears.
2. if a file with the same name already exists on the PC. If yes, the process is aborted with an error message.

If a file with the same name is to be overwritten, select 'Overwrite'. 'Cancel' closes the dialog without copying files.

 **The automatically entered name cannot be changed.**

Elementary functions

The copying process is documented in the following window:



After the process is completed, the total size of the file is indicated under 'Bytes'. Under 'Status' the correct execution and if existing error messages or an abortion are logged.

'Close' interrupts the copying process. Only the current file is transferred completely.

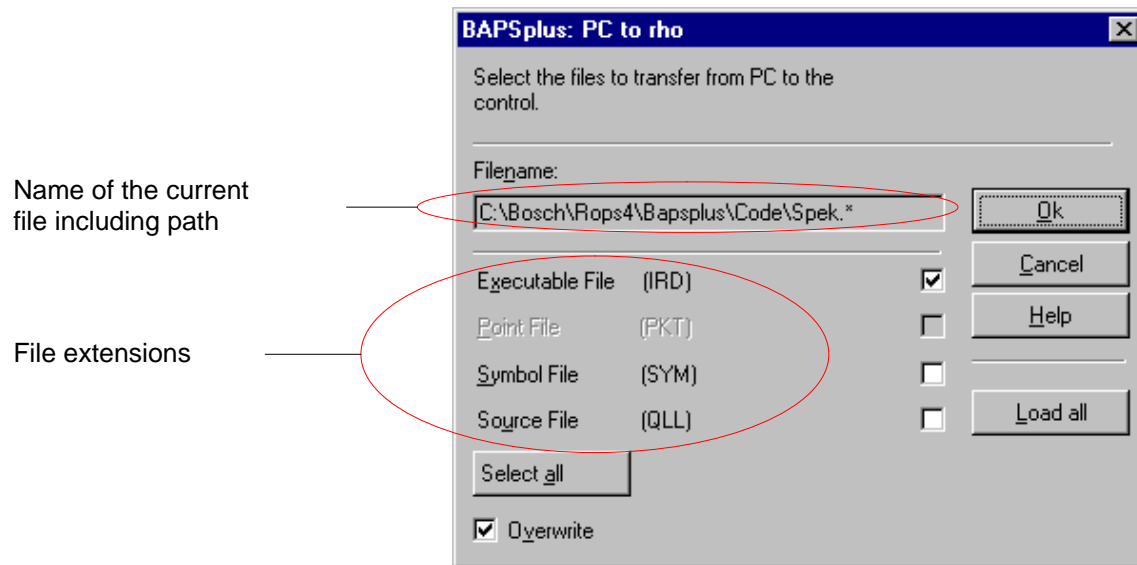
'Dest.' indicates in which directories the corresponding files are copied. The target directories are defined through the file extension, see chapter 3.1.1.

 **<F4> opens the loading dialog directly.**

Copy PC → rho...

'Copy PC → rho...' starts the copying of files from the PC into the control. A dialog to select the files that are to be copied appears:

Elementary functions



Name of the current file including path

File extensions

- ★ Click on desired file extension.
All file extensions are selected via 'Select All'.
- ★ Start copying with 'Ok'. BAPsplus checks
 3. if the desired file exists on the PC. If not, a corresponding message appears.
 4. if a file with the same name already exists on the control. If yes, the process is aborted with an error message.
 If a file with the same name is to be overwritten, select 'Overwrite'. 'Cancel' closes the dialog without copying files.

 **The automatically entered name cannot be changed.**

The copying is documented in an analog window for 'Transfer files from the rho', see page 4-29.

'Source' indicates from which directories the corresponding files are taken. These source directories are defined through their file extensions, see chapter 3.1.1. 'Source' contains no path since the file name is stated on the control.

 **<F5> opens the loading dialog directly.**

4.2.8 Main menu item 'Extras'

'Extras' contains functions

- for searching for incomplete program symbols
- for starting the data entry manager and compiler

Elementary functions

- for editing tools
- for editing commands

After selecting the menu item, the following submenu appears:



Examine Parameters

Most of the symbols in the symbol bar require auxiliary information to enable an error-free compilation of the automatically created program. These data are entered within the parameter dialog which is different for each symbol. If a symbol is sufficiently parameterized, this is indicated in BAPSplus with a small hook.



'Examine Parameters' is now used to scan the program flowchart for incompletely parameterized symbols. All program levels and defined sub-routines starting from the program start are scanned. The first symbol is activated which requires the input of additional data.

With **<Shift+Enter>**, the parameter dialog can now be opened. The missing data can be entered. When all necessary data are available, a corresponding message is displayed.

Start DIM...

'Start DIM...' starts the Data entry manager with the current OPD file. The data entry manager makes an automatically generated surface available for modifying selected program variables, see chapter 6.

Compile...

'Compile...' calls explicitly the compiler integrated in BAPSplus. It will be normally automatically called by the system when the entry dialog is to be closed.

Elementary functions

When it is automatically called, only the first error found in the source program is displayed. When manually called, a list of all contained errors is displayed.

☞ **BAPSplus expects the compiler to be located in the directory entered in 'Options/Directories...'**

☞ **Shortcut: <F6>
Corresponding symbol in the tool bar:**



Edit tools...

Tools can be created and edited with 'Edit tools...', see chapter 5.9.

Save tools...

The tools created with 'Edit tools...' can be saved in a file with 'Save tools...'

After selection, a dialog for entering the file name appears. The name "TOOLS.DAT" is automatically entered, but can be changed if necessary.

☞ **It is possible to create different tool files, however the control expects the tools to be contained in the file TOOLS.DAT. This means that for use on the control, the file name has to be changed correspondingly before transfer, see section 5.9.**

Edit commands...

'Edit commands...' opens a dialog sequence for the definition of new program instructions by which the system can be enlarged and adapted to a further development of the control.

It is possible to cancel new instructions or to adapt already existing instructions to the individual requirements. Recurrent instructions can be included in macros and then used like a single instruction. See section 5.7.

Redraw window

BAPSplus redraws the screen content.

Elementary functions

4.2.9 Main menu item 'Options'

'Options' enables to adapt BAPSPplus to the individual conditions of the system environment used and to the individual preferences of the user.

After selection of the menu item, the following submenu appears:




Check BAPS Sourcecode

'Check BAPS Sourcecode' switches the automatic program test in the background on and off again. If the program test is active, made visible by a hook before the menu text, the created source text is checked at each closing of an entry dialog by the integrated compiler, see section 5.2.

Include comments

BAPSPplus enables to accompany the symbols in the PFC with comments. After activation of the menu item, these comments are also included into the BAPS source text.

 **Comments in the PFC are entered into the attribute dialog of the corresponding symbols. This dialog is available via 'Item/Properties...' with <Alt+Enter> or via the mouse menu 'Properties'.**

Mark BAPS Sourcelines

If this menu item is activated, the current program line is marked in the source text window.

Elementary functions

Show Linenumbers

The line numbers in the source text window can be switched on or off.

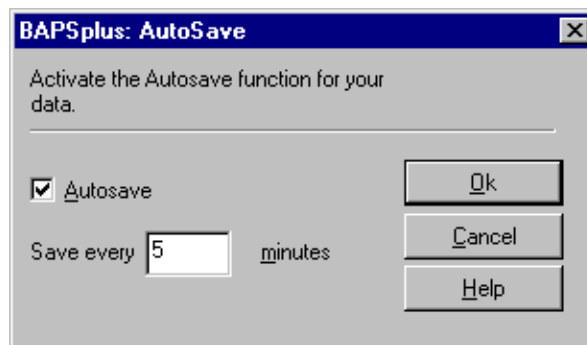
Follow Runposition

When this menu item is selected, the position is shown even in the sublevels in the debug mode. The level in which the current run position is, is always displayed.

Autosave...

Modifications of files can be secured by saving a temporary copy of the file regularly while editing.

To be able to restore the file and the changes that have been performed e.g. after a power failure, the following settings have to be made under 'Autosave...':

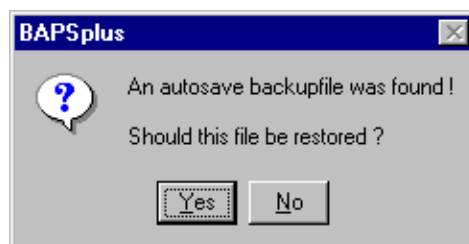


- ★ Click on 'Autosave' to save the files automatically.
- ★ In the field 'Save every n minutes' enter the time interval after which BAPSplus should save the opened files.

Files that are automatically saved are saved in a special format at a certain place until the actual saving process.

When BAPSplus is re-started e.g. after a power failure and the file last edited is called, the system recognizes that a backup exists.

A dialog appears enabling to load the backup and restore the performed modifications:



With 'No' the automatic backup is deleted. After the restored file has been saved, the autosave backup file is removed.

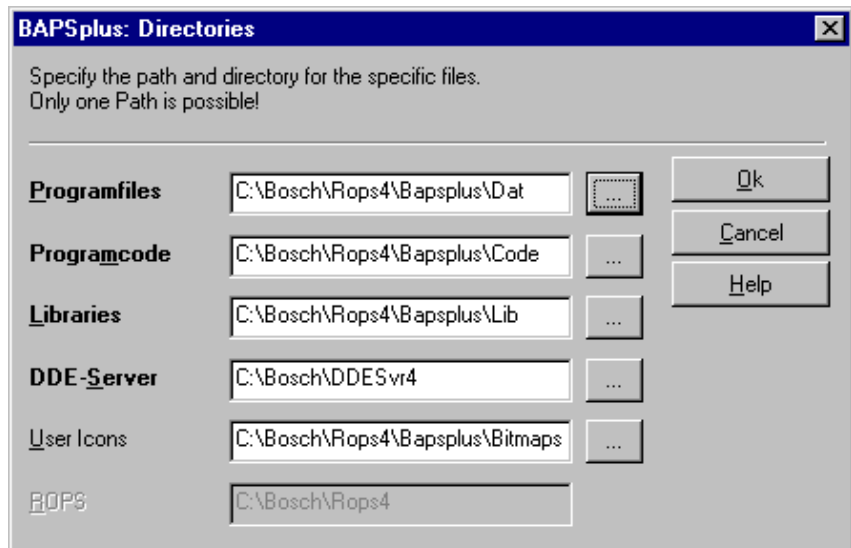
Elementary functions

Directories...

'Directories...' enables to define paths to save the different data types.

It is also possible to adjust:

- compiler directory
- directory in which the DDE server is installed



CAUTION

It is not allowed to use long path names.

They are not supported by BAPSplus and can lead to unforeseen reactions.

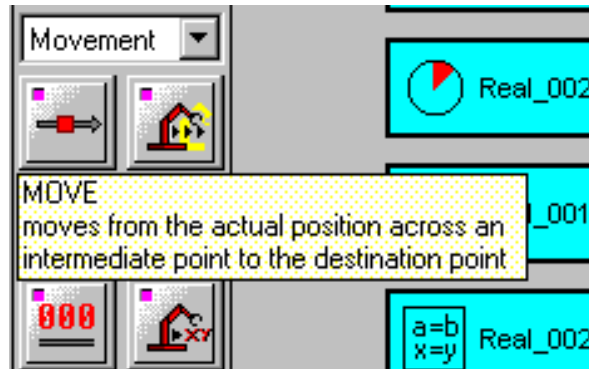
Elementary functions

Quickinfo

The tool bar and the icon bar support quick infos. These are small information windows appearing below a symbol on the right when the mouse pointer remains on this element for a while.

The menu item 'Quickinfo' allows to switch on and off these information windows.

Example:
MOVE instruction



Quickinfos for the tool bar are strictly defined by BAPSplus. The quick infos appearing in the icon bar represent the name of the corresponding icon. The name is entered when an icon is integrated into the icon bar, see page 4-32.

- ☞ **Quick infos disappear automatically as soon as the mouse pointer is moved or a keyboard action is carried out.**

Iconbar

'Iconbar' enables to influence the learning mode of the icon bar. BAP-Splus remembers the four most frequently used symbols and makes them available at the upper rim of the icon bar for the direct access, see section 5.6.

The iconbar is updated as long as the learning mode is active.

After selection, the following submenu appears:



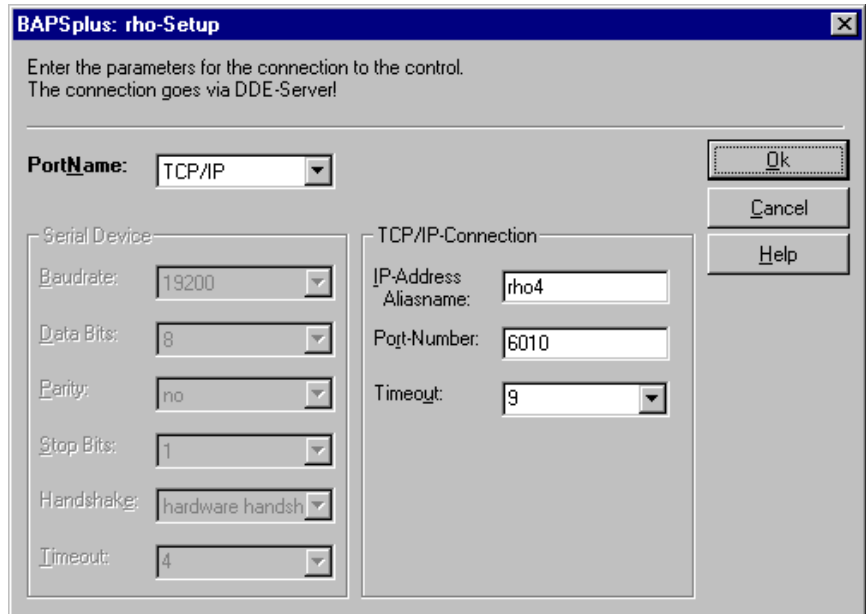
- Learning : switches the learning mode on and off
hook indicates active learning mode
- Reset : resets learning mode

Elementary functions


rho-Setup...

'rho-Setup...' allows to configure parameters for the connection to the control.

The following dialog appears:



The data are read for the program start from the file ROPSWIN.ini.

 **Changes carried out in the dialog are only active during the current session. If the connection data are to be changed permanently, the setting must be carried out in the program Online (Options/ Setup).**

Fonts

The fonts used by the programming system can be changed with 'Fonts'.

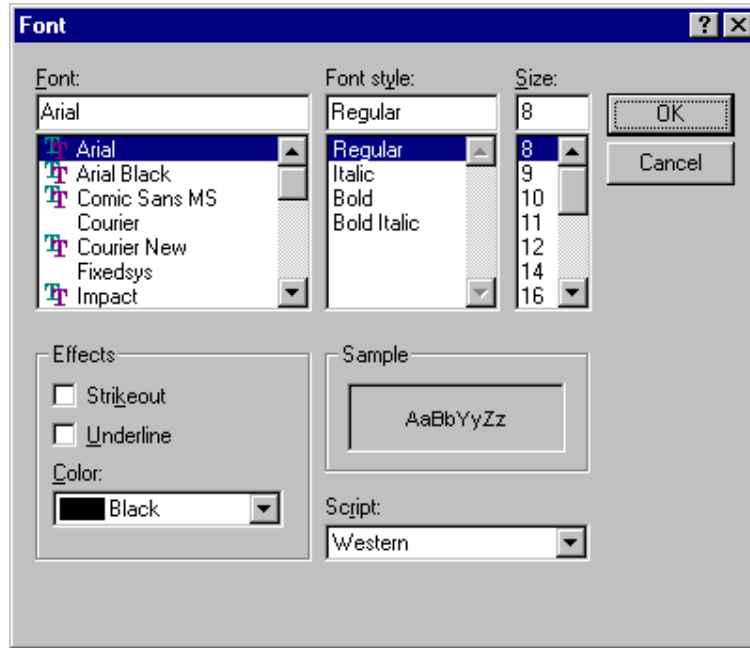
The following submenu appears:



Inside elements...	:	Defines the font used in the symbols of the PFC
Comment...	:	Defines the font used for the comments in the PFC
BAPS source-code...	:	Defines the font used in the source text window

★ Select desired function to start font selection dialog.

Elementary functions



☞ **End selection dialog before continuing work.**

Colors

To change the colors for cursor, breakpoint and run position.

4.2.10 Main menu item 'Tool'

The entries of the submenu can be allocated freely. Program calls required for the work at the PC can be included. These programs can be then directly started from BAPSplus.

Define functions

Subitems are defined in the initialization file BAPSPLUS.ini. The description is entered under [TOOLS].

It contains:

- the name of the subitem which can be freely selected
- the call of the desired program.

To optically separate the subitems, a dash can be inserted in the menu via the instruction 'SEPARATOR='.

☞ **The modifications become active only after a restart of BAPSplus.**

Elementary functions

**Example:
Define user-defined menu item**

When the submenu is shown, you can select:

- 'Editor' via 'E'
- 'Online' via 'O'
- 'MS-DOS' via 'M'

The requirements are met through the following entries in BAP-SPLUS.ini:

```
[Tools]                               ;Gruppe TOOLS
```

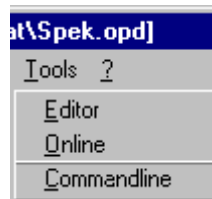
```
&Editor=C:\Bosch\Rops4\Edit.exe
```

```
&Online=C:\Bosch\Rops4\Online.exe
```

```
SEPARATOR=
```

```
&Commandline=dosprmt.pif
```

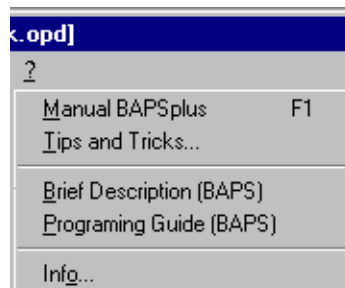
Appearance of 'Tools' after the restart:



4.2.11 Main menu item '?' (Help)

'?' activates the help function and supplies additionally information about the program version and the copyright remarks.

The following submenu appears:



Manual BAPSplus

The manual 'BAPSplus' is displayed via Acrobat Reader. If the Acrobat Reader program is not installed, an error message appears.

Elementary functions

Tips and Tricks...

Useful tips and tricks for using BAPSplus.

Brief description (BAPS)

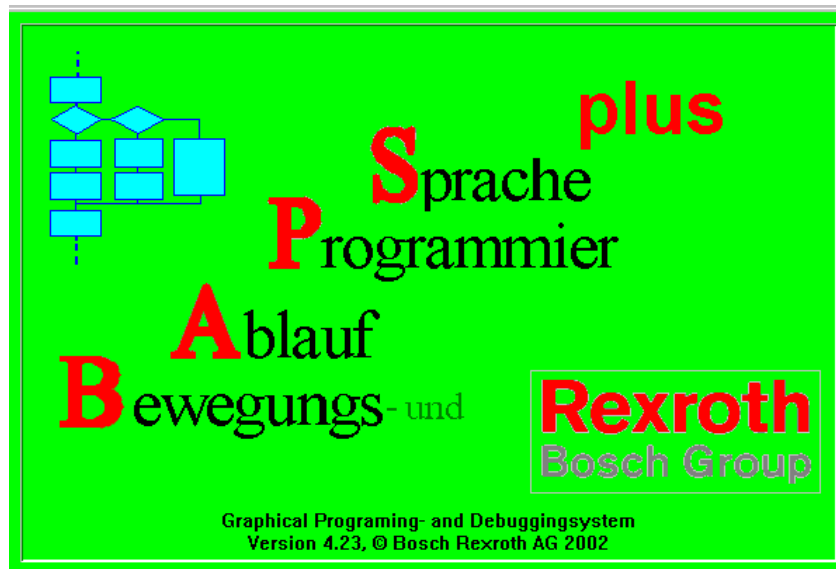
The manual 'BAPS Short description' is displayed via Acrobat Reader. If the Acrobat Reader program is not installed, an error message appears.

Programming Guide (BAPS)

The manual 'BAPS Programming Guide' is displayed via Acrobat Reader. If the Acrobat Reader program is not installed, an error message appears.

Info...

Information about BAPSplus version and the copyright remarks.



☞ The window closes after a few seconds. A mouse click shortens that time.

☞ This BAPSplus start screen is located under OEMINFO.BMP in the icon directory and can be modified or replaced to realize you own start screens.

Expanded Functions

5 Expanded Functions

Expanded functions facilitate the work with BAPSplus and increase the efficiency.

 **This chapter is determined above all for users having already experience.**

5.1 Presettings and program data

Functions for the definition of the default settings (e.g. kinematics, sub-routines) and program data (e.g. constants, variables, signals) are accessible through the context menu (right mouse key, menu item 'Parameter...') of the BEGIN element in the PFC or through the corresponding symbols in the tool bar.

5.1.1 Default kinematics

BAPSplus offers the possibility of re-building the kinematics existing in the real system and offering them for use as standard after each re-start.

 **For detailed information about kinematic and coordinate definition, see manual 'BAPS3 Programming manual', chapter 'Program structure'.**

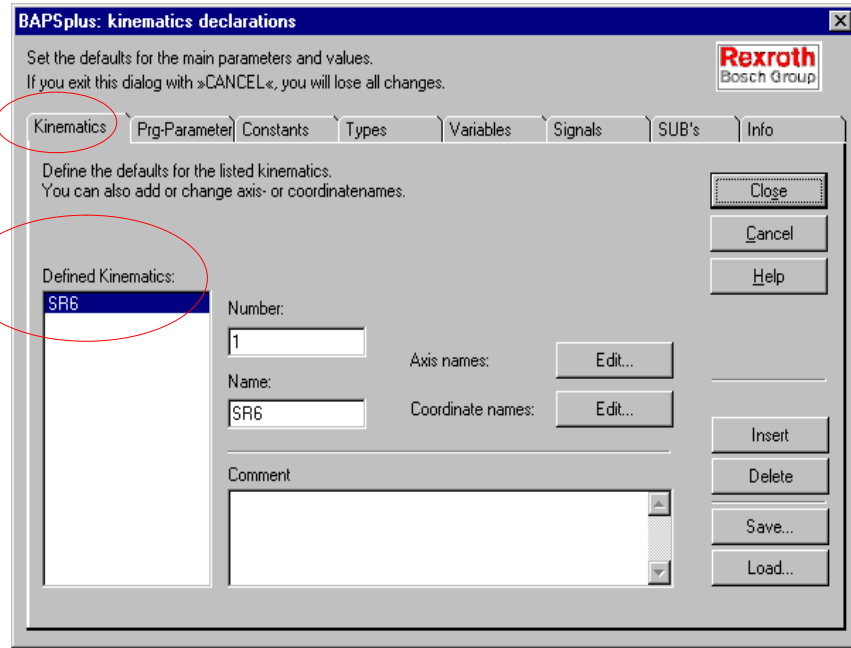
★ Open dialog 'Kinematic declaration' to define kinematics.

 **Corresponding symbol in the tool bar:**



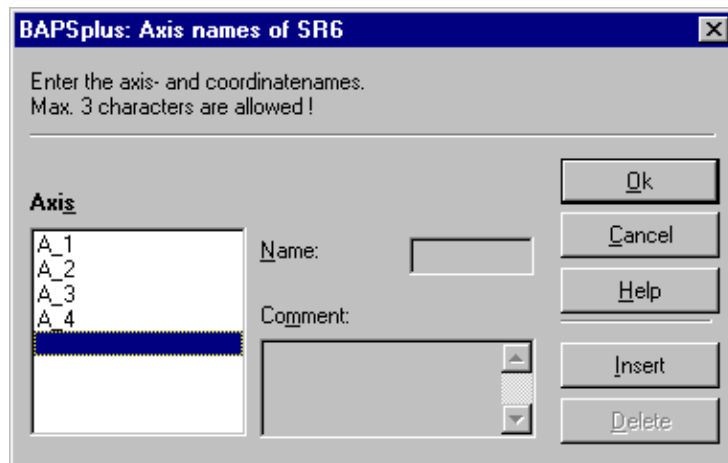
Expanded Functions

All declared kinematics



- ★ Enter number, name and comment in the corresponding text fields.
- New kinematics are created with 'Insert', those that are no longer required deleted with 'Delete'.
- ★ Save permanently kinematics with 'Save' in a file (default setting DEFAULT.okd).
 - ★ Replace kinematics that are currently defined with 'Load' through the kinematics contained in the file.


'Edit...' allows to change axis and coordinate names.



- ☞ The dialog can only be operated when the main program level is active, i.e. is visible in the PFC. The entry elements are locked otherwise.

Expanded Functions

5.1.2 Program declarations

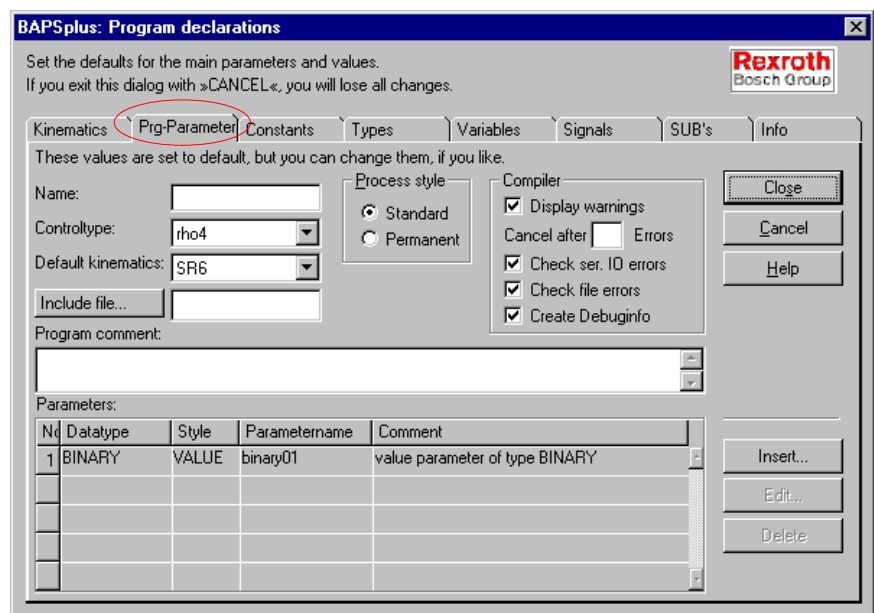
 The dialog 'Program declarations' can only be operated when the main program level is active, i.e. is visible in the PFC. The entry elements are locked otherwise.

★ Open dialog to define program settings.

 Corresponding symbol in the tool bar:



For detailed information, see manual 'BAPS3 Programming manual', chapter 'Program structure'.



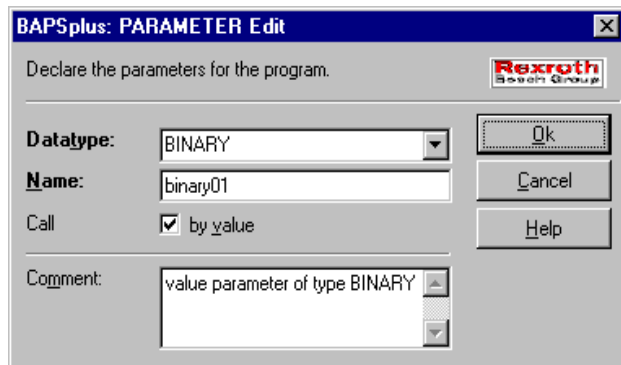
The following parameters can be entered, adjusted and selected:

Name	:	Enter name of the new program. If the field is empty, BAPSplus uses the file name without file extension.
Control type	:	Select current control type
Default kinematics	:	Select current kinematic
Include file	:	Integrate optional include file via file selection dialog
Process style	:	Standard Setting for usual processes in automatic operation
		Permanent executable in the operating modes 'Automatic' and 'Manual', cannot be aborted with 'Reset' etc.

Expanded Functions

- Compiler** : Following settings are possible:
- Display warnings prevents the issue of warnings
 - Cancel after n errors n = number of errors after which the compiler aborts the compiling (n = 1..100).
 - Create debug-
infos Create test information
If this information is not created, the program cannot be tested with the test system.
 - Check ser.
IO errors Prevents the user program abortion when interface errors occur.
 - Check file errors Prevents the abortion of a program when the ASSIGN instruction occurs on opened file.
- Comment** : Enter comment about program.
If a file is used as a template, the beginn text of the comment appears in the selection menu of 'File/ New...', see chapter 5.4.
- Parameter** : The transfer parameters of the current program, if existing, are listed in the corresponding list.

The dialog 'BAPSplus: PARAMETER Edit' appears via 'Insert...' or 'Edit...' for changing the parameters. Parameters in surplus are removed via 'Delete'.



Expanded Functions

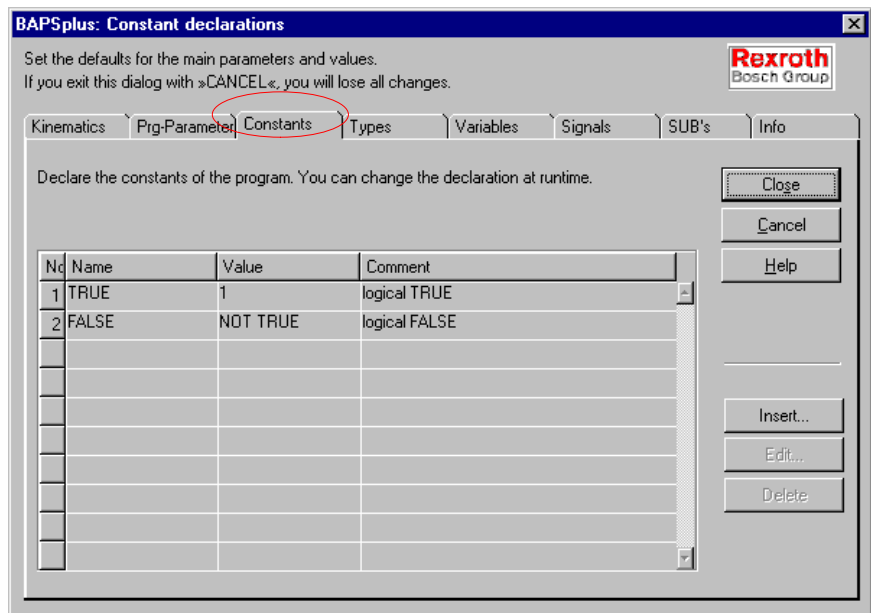
5.1.3 Definition of constants

For detailed information about constants, see manual 'BAPS3 Programming manual'.

☞ **Corresponding symbol in the tool bar:**



★ Perform declaration in dialog 'Constant declarations'.



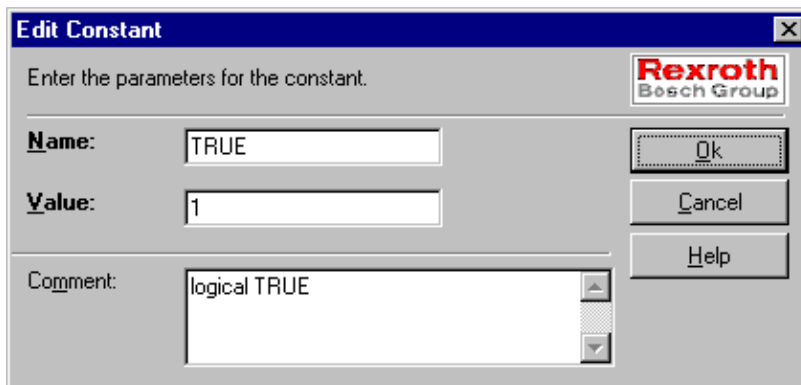
All known program constants appear in the table. The column 'No.' indicates the order in the source program.

☞ **If the constants depend on each other, the correct order must be ensured (See above, constant TRUE is defined before constant FALSE.)**

By clicking the column heading, it is possible to toggle the switching of the corresponding column (increasing or decreasing order).

A dialog for editing the constants opens via 'Insert' or 'Edit'.

Expanded Functions



Possibilities of entry for constants:

- Name
- Initialization value

It can only be a number, a mathematical expression, a character or a string.

If this constant should be used for field limits, only integer constants are allowed or for expressions it must be possible to attribute all parts to an integer value.

- Comment

 **The comment can also be directly changed in the dialog 'Constant declarations'. For this, click on the desired comment field.**

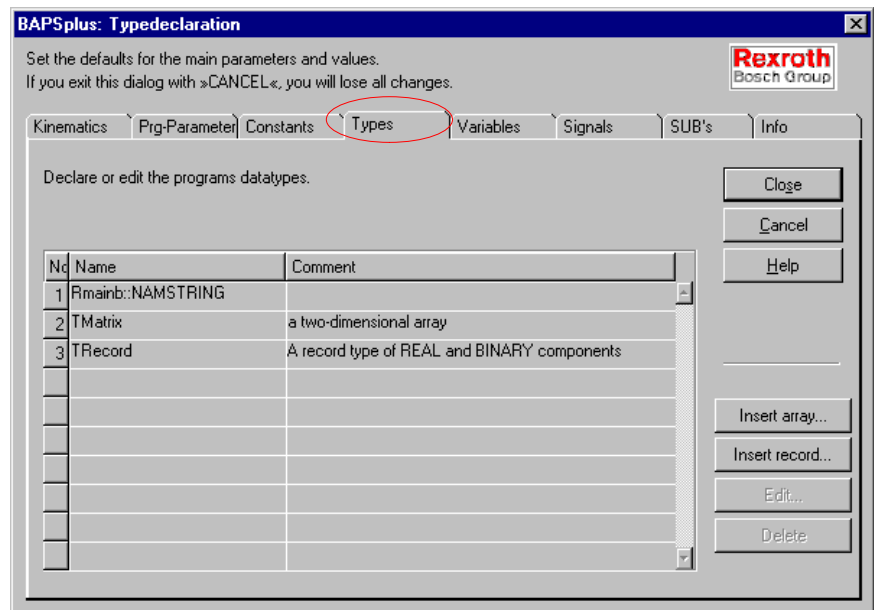
Expanded Functions

5.1.4 Declaration of type

The dialog 'Type declaration' shows all types used in the program. A difference is made between:

- Library type (see figure: Type NAMSTRING from the library RmainB)
- local types (here: TMatrix und TRecord)

Library types are available for each program and administrated dynamically.



☞ In 'BAPSpplus: Type declaration', only library types are listed that are also used in the current program. Not all types declared in the loaded libraries are represented.

Expanded Functions

Declaration of array types

- ★ Click 'Insert array' in the dialog 'BAPSplus: Typedeclaration' to declare array types.

The type of array can be declared with a maximum of 3 dimensions. Constants and mathematical expressions are authorized as array limits as long as the result is integer.

- ☞ **When changing the dimensions, the contents of the initialized variables are not adopted.**
- ☞ **When the element of a limit is a number with a floating decimal point, the result is taken as a floating decimal point number and cannot be used as a limit.**
Example:
' $4*2.5$ ' is not authorized, but ' $4*5/2$ ' is a valid limit.

Expanded Functions

Declaration of record types

- ★ Click on 'Insert record...' in the dialog 'BAPSplus: Typedeclaration' to declare record types.

BAPSplus: RECORD Declaration

Enter the components of the recordtype.

Name TRecord

Comment A record type of REAL and BINARY components

No	Type	Componentname
1	REAL	m_REAL
2	BINARY	m_BIN

Buttons: Insert..., Edit..., Delete..., Ok, Cancel, Help

Components are added, deleted or edited via the corresponding buttons.

- ☞ **There is no limitation for the number of components.**

All types previously defined and library types can be used as types. It must be taken into account that the declaration of library types in the BAPS source program is inserted into the main program even if the library type is used in a subroutine.

The components can be edited either directly in the corresponding editing field or in the following dialog:

BAPSplus: Record Components

Edit the type and the name of the componenten.

Componenttype BINARY

Componentname m_BIN

Buttons: Ok, Cancel, Help

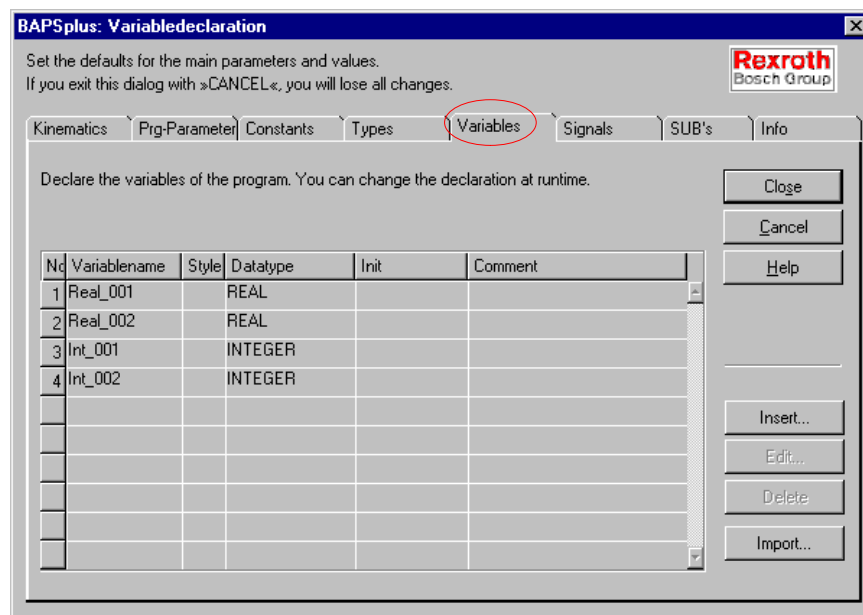
Expanded Functions

5.1.5 Declaration of variables

The variables of the program are declared via the register card 'Variables'. The listing is represented in a table form. The sorting is defined through mouse click on a column heading.

The columns 'Init' and 'Comment' can be directly changed with a mouse click.

 **Corresponding symbol in the tool bar:**



'Import...' opens a file selection dialog to define the file to be exported. After completed selection, another dialog appears in which the variables to be imported can be marked.

Initialized variables have a special status, since the corresponding type may have been deleted or changed. The button 'Edit' takes over the control (in the variable declaration dialog and in the parameter dialog of assign elements).

A distinction is made between:

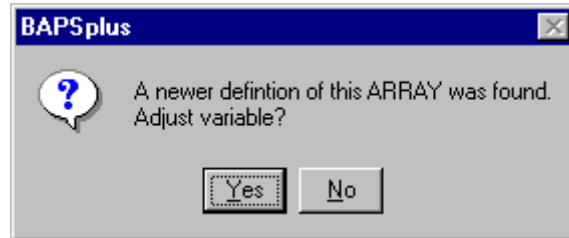
- Extension of the existing type, e.g. larger field or new component
- Reducing of the existing type, e.g. smaller type or deletion of a component.

In the first case, all existing data is adopted, in the second, the data must be deleted so that they correspond to the new type.

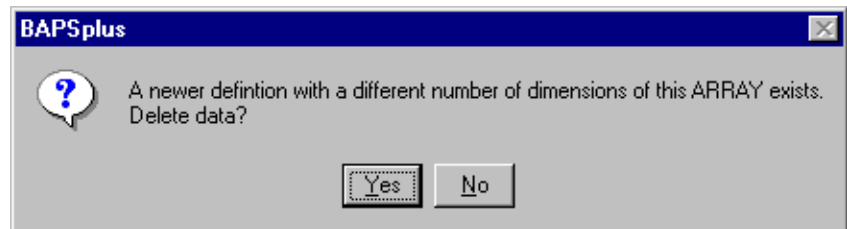
Expanded Functions

With fields of the same dimension, the remaining fields are filled with data, with types of record, a dialog is opened to adapt and/or complete the components. The worst case that can occur is a change from RECORD to ARRAY or vice-versa since these data cannot be adopted.

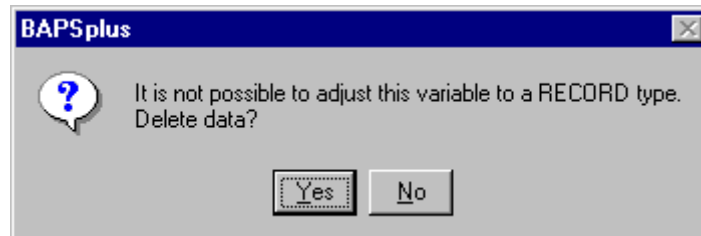
Before an array variable is changed, a safety inquiry appears:



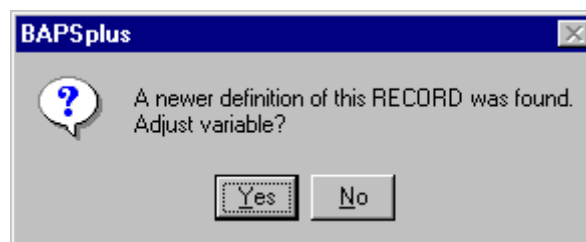
Before data is deleted in the case of a dimension change, a safety inquiry appears:



Before data is deleted since the type is now a 'Record', a safety inquiry appears:

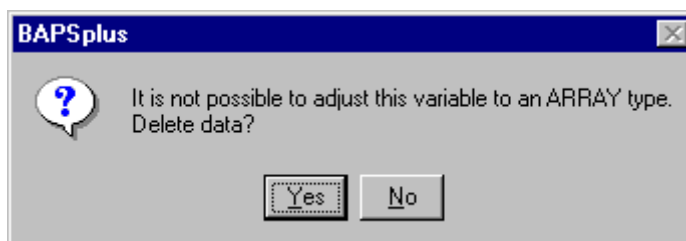


Before a record variable is changed, a safety inquiry appears:

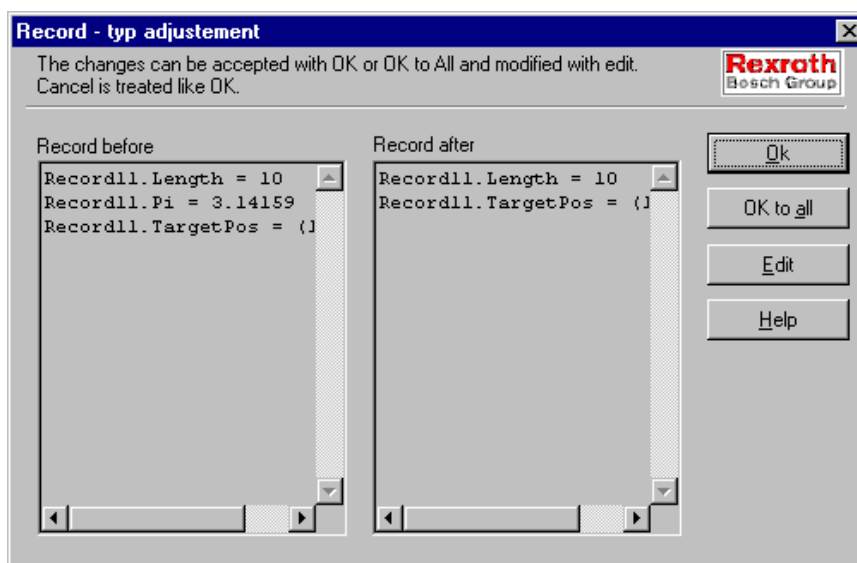


Expanded Functions

Before data is deleted since a conversion from RECORD to ARRAY is not possible, a safety inquiry appears:



Record types are modified via the following dialog:



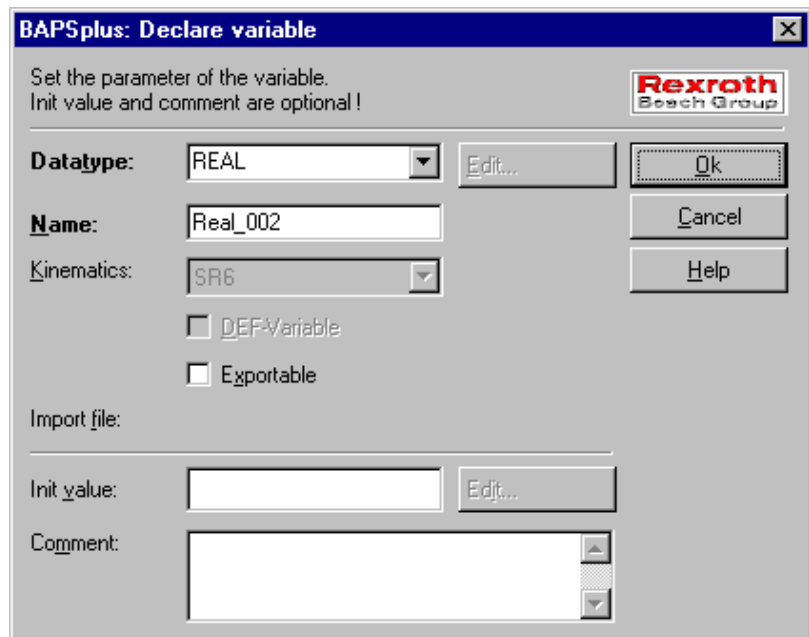
No dialog appears:

- for arrays since they cannot only be copied into the new one with corresponding limits.
- in case the left and right pages are identical.

Expanded Functions

'Insert' and 'Edit' in the dialog 'BAPSplus: Variabledeclaration' lead to the dialog 'Declare variable'.


In this dialog, the properties of the variable can be exactly specified:



- Select data type from the list with all valid data types.
'Edit' is active when a field or data type is selected as data type. (See 'Declaration of array types', page 5–8 or 'Declaration of record types', page 5–9). 'ARRAY (new)' and 'RECORD (new)' enable to create a new array resp. a new record.
- Enter variable name:
When the text field is selected for the first time, the programming system generates a name suggestion (consisting of shortened type name and a consecutively running number). This name must not be longer than 12 characters.
- Select corresponding kinematic:
This entry possibility is valid only for point variables (POINT or JC_POINT) or fields of points.
- Mark checkbox 'DEF-Variable':
Determines that the variable is a teach variable. This selection field is also only accessible for point variables and fields of points.
- Mark checkbox 'Exportable':
Exports the variable as a global variable.
- Line 'Import file':
For the case of imported variables, the file name, from which the variable is exported, appears here.

Expanded Functions

- Initialization for program beginning (initial value is optional):
Initialization instructions are inserted in the BAPS source program directly after program start. No symbol appears for it in the program flow chart.
- Remark about the variables:
This comment is adopted in the source text when the menu item 'Options/Comment' is active.

 **The character '@' need not to be entered with variables of type JC_POINT. It is automatically set.**

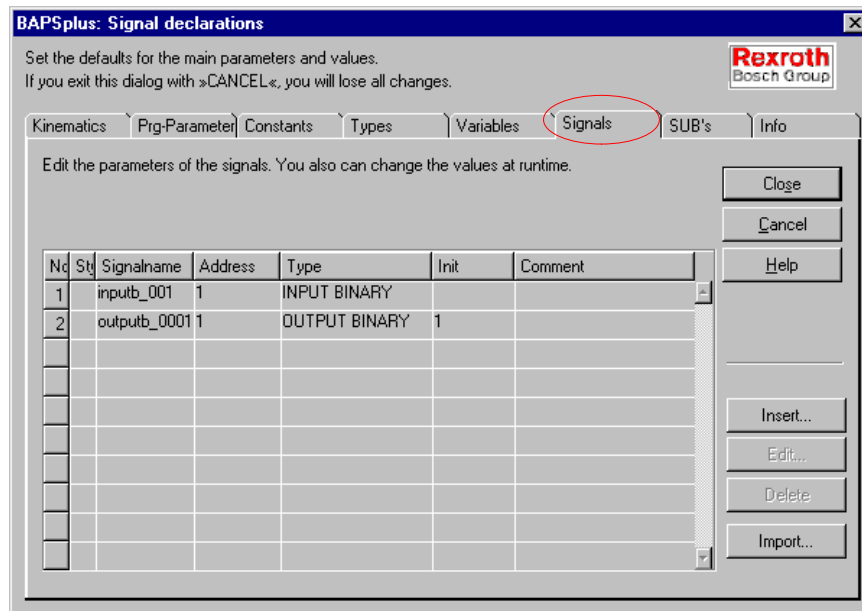
5.1.6 Define signals

The signals of the program are declared via the register card 'Signal declarations'. The listing is represented in table form and the sorting can be defined through mouse click on a column heading.

The columns 'Init' and 'Comment' can be directly changed with a mouse click.

 **Corresponding symbol in the tool bar:**





'Import' opens a file selection dialog to define the file to be exported. After selection, another dialog appears in which the signals to be imported can be marked.

Expanded Functions

The operation is the same as with the dialog 'Variable declaration', see section 5.1.5.

In the dialog 'Edit signals', the signals are defined:

- Select signal type from list of all known signal data types
- Enter signal name:
When the text field is empty, the programming system generates a name suggestion (consisting of the shortened type name and a consecutive number). The maximum length for signal names is 12 characters.
- Select corresponding address:
All known constants are offered for selection in the combo box. The direct entry is also possible.
- Mark checkbox 'Exportable':
Exports signal.
- Line 'Import file':
For the case of imported signals, the file name, from which the signal is exported, appears here.
- Initialization for program beginning ('initial value' is optional):
Initialization instructions are inserted in the BAPS source program directly after program start. No symbol appears for it in the program flow chart.
- Remark about the signal:
This comment is adopted in the source text when the menu item 'Options/Comment' is active.

Expanded Functions

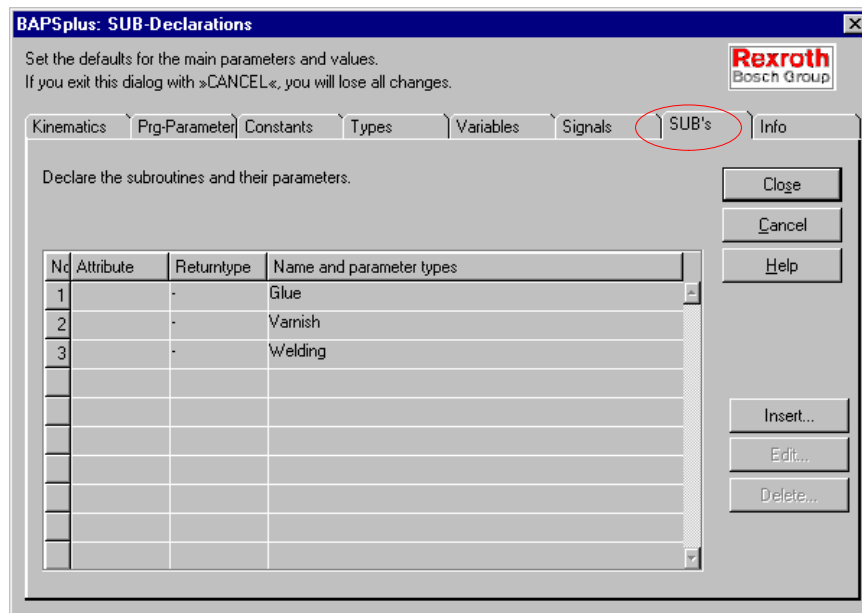
- ☞ For additional information about channel types and their numbers, see manual 'BAPS3 Programming manual', section 'Channels'.

5.1.7 Define subroutines

- ☞ The dialog 'SUB declarations' can only be operated when the main program level is active, i.e. is visible in the PFC. The entry elements are locked otherwise.

The program subroutines are declared via the register card 'SUB's'. The listing is represented in table form and the sorting can be defined through mouse click on a column heading.

- ☞ Corresponding symbol in the tool bar:



The subroutines can be declared in any order.

- ☞ For additional information about subroutines, see 'BAPS3 short description', section 'Subroutine declarations'.

Expanded Functions

'Insert' or 'Edit' lead to the following dialog, in which the subroutine is edited:

BAPSplus: SUBROUTINES

Enter parameters of the subroutine.
You must specify the name, other statements are optional.

Name: **Return type:**

Style:

Subroutine External program
 rho function Special function **Number:**

Parameter:

Nr	Datatype	Style	Name	Comment
1	REAL		c_intensity	welding current

Comment

Define parameter dialog...

Subroutine Color...

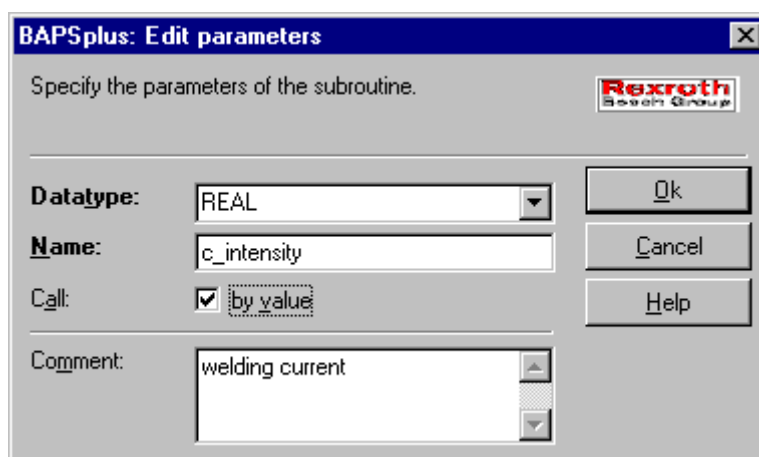
Insert...
Edit...
Delete

It offers the following possibilities:

- Input the subroutine name.
- Define the type of subroutine.
 - local subroutine
 - external main program
 - rho function (define number and return value)
 - Special function (define number)
- When parameters have been determined, the form of the dialog in which these parameters are inquired when being called in the program flowchart, can be determined in the 'Define parameter dialog...' switch field.
- Select the color of the PFC symbol for the call of the subroutine

Expanded Functions

- Listing of the transfer parameters. The entry is made in a separate dialog.



- ★ Click on checkbox 'By value' when the change of the parameter should be transmitted within the subroutine to the caller. If the checkbox is marked, the variable that should be transmitted to the subroutine has the same value as before being called after the return from the subroutine – even if the variable is changed in the subroutine.
- ★ In 'Comment' enter explanation about subroutine. They are adopted in the source text when the menu item 'Options/Include comments' is active.

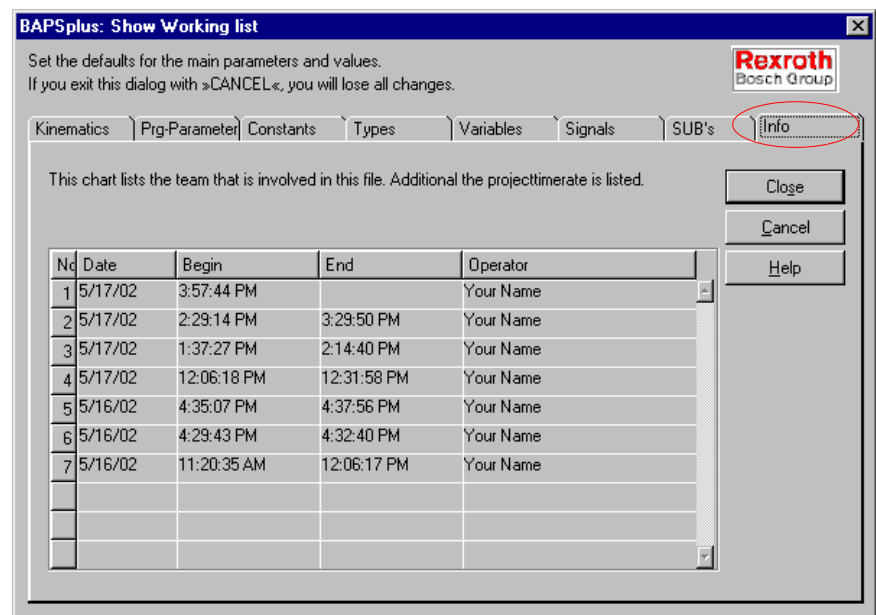
Expanded Functions

5.1.8 Display change history

The change history is represented with the register card 'Info' in the dialog 'BAPSplus: Show Working list'.

The listing is represented in table form and the sorting is represented with mouse click on a column heading. An edition is not planned since the logging is made automatically by BAPSplus.

☞ **Corresponding symbol in the tool bar:**



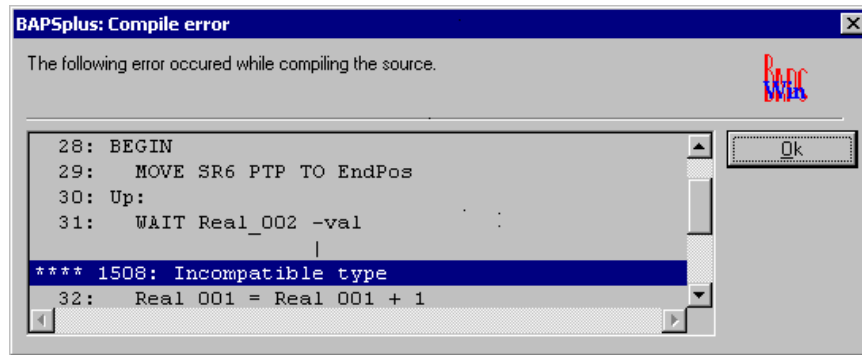
Expanded Functions

5.2 Incremental Compiler

By the automatic source text generation of BAPSPplus it is ensured that the program structure is syntactically correct. This, however, does not apply to the parameters entered by the user.

To check parameters for their correctness, an incremental compiler that directly checks every entry has been integrated. When detecting an error, the error will be issued in plain text and the corresponding parameter dialog opened again.

- ☞ **The automatic program test can be switched off and on again via the menu item 'Options/Check BAPS Sourcecode' when required. The program test is activated in the default setting.**



So it is guaranteed that the dialog boxes for the entry of parameters can only be left after having entered all required parameters correctly.

The parameter dialog can be left with 'Abort' even if erroneous data was entered.

- ☞ **The error text appears after closing the message window in the status line.**

Expanded Functions

5.3 Test programs

After starting the debug mode via the menu item 'Debug – Select program...' or button in the tool bar), it is tested whether a connection to the control has been established. Otherwise a corresponding message will be displayed. The programming mode remains active.

If the file has been changed in the program flowchart, the question is asked, whether the current program is to be translated and loaded into the control. If the file being displayed is a new file (file name 'NON-AME.opd'), the desired file name can be entered before saving the file.

 **If a file already loaded into the control is to be tested, the corresponding .opd file has to be loaded at first.**

BAPSplus checks after selection of the file to be tested if the executable file (file extension '.ird') already exists on the control. Otherwise, after inquiry of a dialog with abort possibility, the file is loaded from the PC into the control.

If the files are not identical, this is also indicated by a corresponding message. The program can either be loaded directly into the control or tested. The latter is e. g. preferable if only smaller changes have been made in the program and if large files with long loading times are concerned.

After correct selection of the test mode, the program is started with 'Debug – Start/Continue program'. Then, the program code window is opened and the current program line is marked. Also in the program flowchart the program can be pursued simultaneously.

For a more detailed diagnosis, it is within the program possible to set breakpoints, and to display and edit variables.

The debug mode is terminated by selecting the menu item 'Debug – Stop debugging', resp. it ends automatically after having reached the program end.

5.4 Working with program masters

Every program file (file extension '.opd') can serve as a template. For this, it has only to be copied into the subdirectory 'Templates'.

Since BAPSplus creates this submenu only when the application is started, a program start is necessary so that newly created templates under 'File/New' are available.

Expanded Functions

The comment for the master file, entered in the 'Program declaration' (see section 5.1.2) serves as a menu entry. If this comment is missing, the file name of the template will be used.

5.5 Working with libraries

An important advantage of BAPSplus is the ability to be expanded and adapted to both the applications of the user and the future abilities of the robot control.

In addition to special function libraries and programs, new control commands must be frequently developed at short notice for customer specific requirements. That is why it is possible to extend BAPSplus with component libraries – composed of subroutines and functions – and completely new control commands.

These functions and commands are created interactively in the system and linked with symbols at will. The appearance of the corresponding dialog boxes is here also variable. If desired, your own input masks with graphics etc. can be created and integrated.

The program code is also generated automatically for new control commands. After including them, the user can no longer distinguish between standard BAPS commands and new commands and/or functions from libraries.

New libraries are created by selecting the menu item 'File/New/Library'. They can be edited like normal program files with the exception that no instructions can be inserted in the main program section.

Normal program files cannot be saved as libraries.

New commands are generated via 'Extras / Edit commands...', see section 5.7.

Expanded Functions

5.6 Working with the icon bar

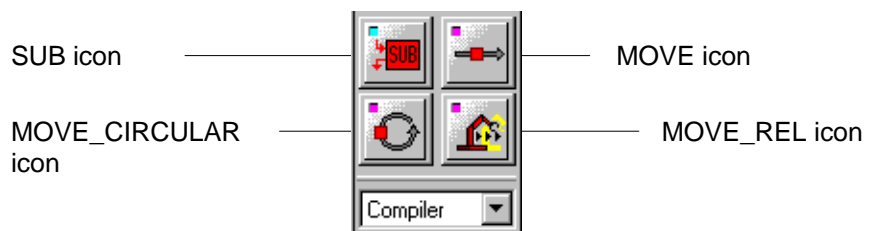
The icon bar incorporates an integrated learning mode which can be influenced by the menu item 'Options/ Iconbar'.

After the installation, the learning mode is active as a standard, but can be switched off if required with 'Options/ Iconbar/ Learning'. In this case, the hook in front of the menu entry disappears.

Whenever inserting an icon from the icon bar into the program flowchart, the programming system is accompanied by a counter. By evaluating this counter, the system determines the order of the most frequently used symbols. During the programming, the learning field of the icon bar is automatically updated, normally the icons of the learning field change at the beginning more frequently.

The learning field contains the four most frequently used symbols, which can be from different icon bars. They can be selected directly.

Exemple of learning field:



'Options/Icon/Reset' enables to reset the learning mode into the state after the installation.

5.6.1 Icon grammar

The icons in the icon bar are created according to grammar rules. Knowledge about it helps to understand the symbols and better recognize them again.

Background

A stylized 'metal button' serves as a uniform background.

The colored spot in the left top corner color-codes and clearly separates the individual areas from each other in addition to the group symbol (movement commands, compiler instructions, standard functions, ...).

Expanded Functions



Group symbols

The commands to be created are often abstract terms. It is not always easy to find icons with a corresponding image being self-explanatory for the viewer.

To make a structuring easier, a background symbol, the so-called group symbol, has been used apart from the color code, which separates specific groups from each other.

Example:

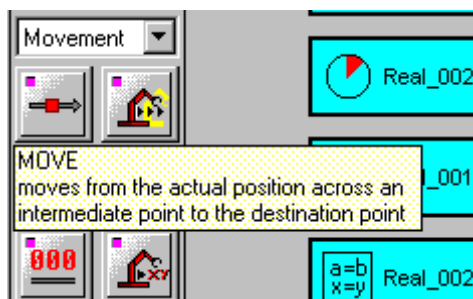
For the compiler instructions, which influence for example the working method of the compiler, a stylized funnel has been used. It symbolizes a compression of the extensive source text into the compact format of the executable file. Whenever the user finds a symbol with a funnel, he knows immediately that this is a compile instruction to the compiler. For the creation of new symbols this means furthermore that it is not always necessary to find a new image for the term 'compiler', but only for the specific compiler function.



Compile instruction 'Select interpolation type' where 'INT' visualizes the function 'Interpolation'.

Basic symbol
Compiler instruction

'Quickinfos' and 'WinTips' also make the assignment easier. They are information texts appearing when the mouse cursor stays for a certain time of the symbol. They can be switched off via an option.



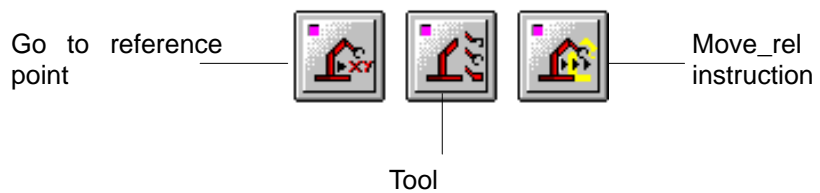
Expanded Functions

Coloration

For the coloration the colors provided by Windows have been used with soft tones. The color, which is the most striking in the icon, indicates the main points of the icon function.

Similar icons

For similar or slightly changed processes, the same source materials have been kept. For all icons referring e.g. to the real robot, a stylized robot is used. This makes it easier for the user to deduce the function behind the icon.



Experience

When creating symbols, one should always be aware that the extent of interpretation increases with the reduction of the presented information, and so is increased also the danger of misinterpretations.

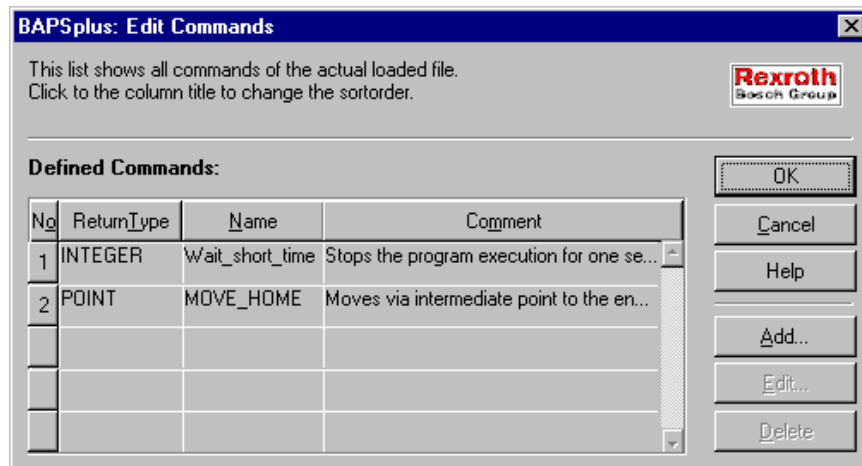
☞ **However, experience shows that the user becomes familiar with even abstract symbols when the functions are frequently used.**

Expanded Functions

5.7 Working with new commands

The mode for editing new commands is activated with 'Extras/ Edit commands...'.
 .

A dialog appears which lists all commands contained in the loaded program or library. New commands are here adopted with 'Add...', modified with 'Edit...' and removed with 'Delete'.

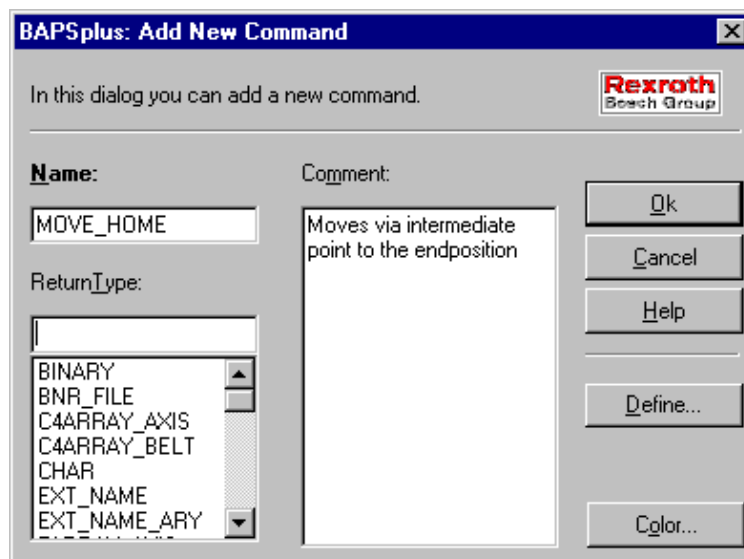


Dialogs for the definition of new control commands or macros

Example:

Implementation of a new command which moves the robot to the deposit position via an intermediate position that can be freely chosen. The command is based on the MOVE instruction.

The editing of new commands is divided into two dialogs:



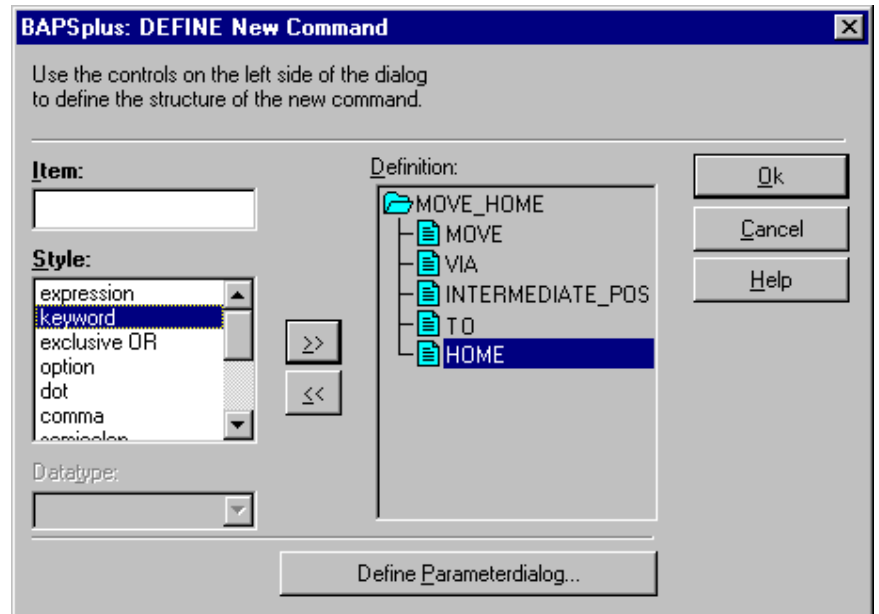
Expanded Functions

- ★ Enter name, return data type of the new command (optional) and additional comment.

The comment is displayed when the new command in the PFC is selected in the status line.

- ★ To define the actual command structure, select 'Define...'

A dialog appears in which the actual command structure is determined in form of a hierarchy tree:



- ★ Enter name of the element (or element itself in case of a hyphen).
- ★ Select data type (optional).
- ★ Select type of the element (hyphen, key word ...) from selection list 'Style'.
- ★ Click on '>>' to include the element into the definition structure underneath the current cursor position.

Command order in the example:

- ★ Enter MOVE
- ★ Select 'keyword' style
- ★ Click on '>>'
- ★ Enter VIA
- ★ Select 'keyword' style
- ★ Click on '>>'

Expanded Functions

- ★ Enter 'Intermediate point'
- ★ Select data type 'Point'
- ★ Select 'expression' style
- ★ Click on '>>'
- ★ Enter TO
- ★ Select 'keyword' style
- ★ Click on '>>'
- ★ Enter HOME
- ★ Select 'keyword' style
(the value of the input field 'Item' is to be taken over unchanged into the structure, and thus into the BAPS source text)
- ★ Click on '>>'

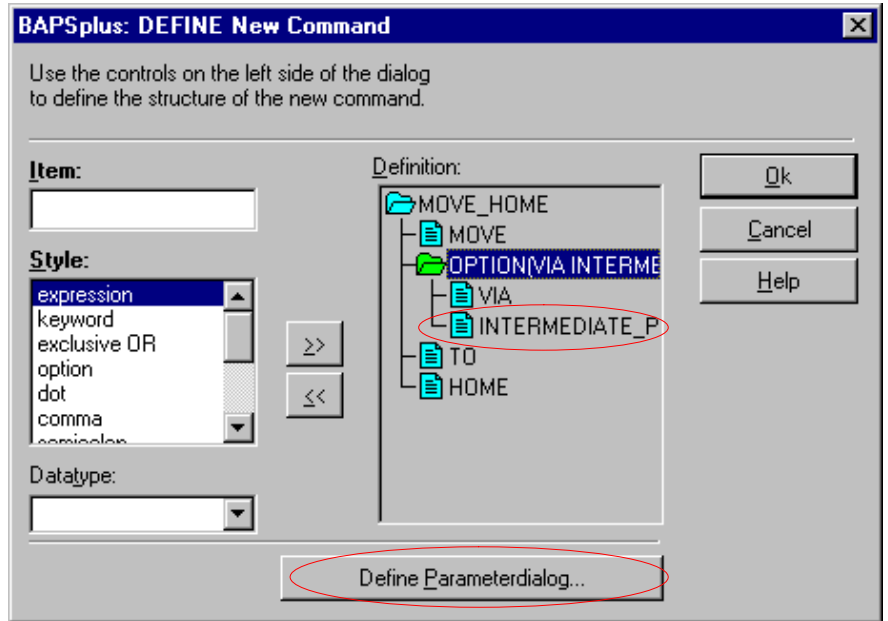
With the '<<' key, an element marked in the right list will be removed from the hierarchy tree and adopted in the left column for editing. It can be changed there and subsequently re-inserted into the hierarchy tree.

In the same way, an element can be deleted from the hierarchy tree (remove without inserting again).

The element type 'Option' contains further elements. If these elements also offer options, they can be opened or closed by a mouse click (in a similar way as under the Windows File Manager or Explorer). 'Options' are inserted and deleted by means of the '>>' and '<<' button.

If it should not be compelling to enter e.g. the intermediate position, it must be declared as an option.

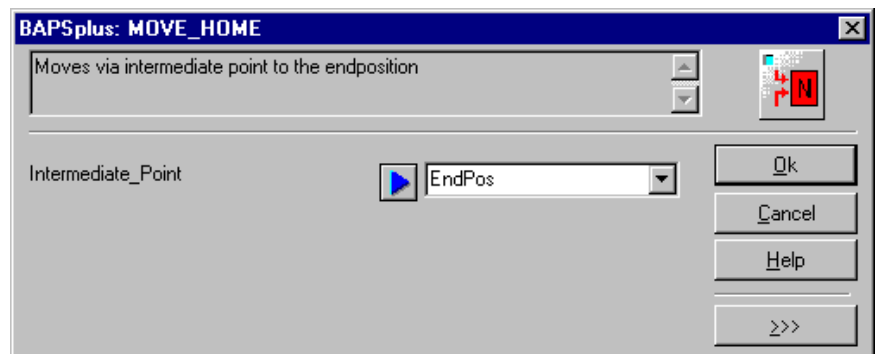
Expanded Functions



- ☞ **An optional definition without expression (variable element) is not represented. This applies to all levels.**

The appearance of the parameter dialog for the new control command can be defined via 'Define parameter dialog...'

'Define parameter dialog...' is only available when the command to be edited contains at least one element of the type 'Expression'. BAPSplus arranges the entry elements of the parameter dialog automatically.



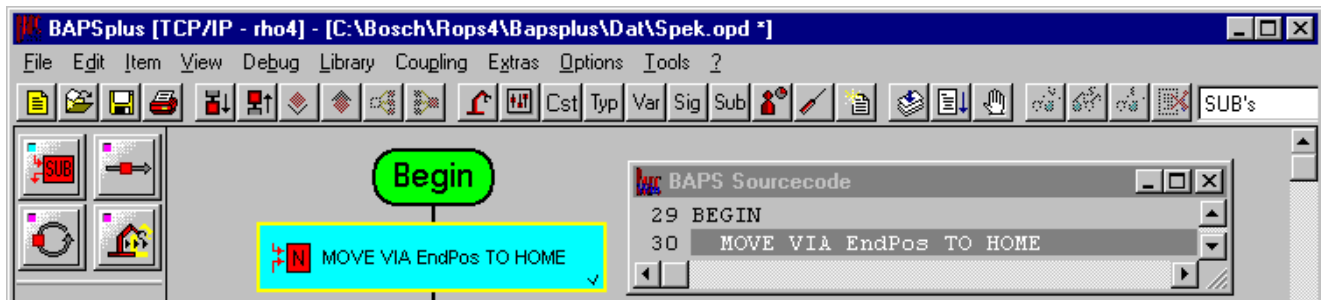
- ★ Click on icon as represented in the margin to use command in the PFC.



A parameter dialog appears which makes all known new commands available for selection.

Expanded Functions

Example:
Command 'Move_Home' including code window




Expanded Functions

5.8 Use export library files

In export library files, all commands and library functions used in the current program are saved.

If a file is to be developed further in another system, an export library has to be created via 'Library/ Export...' and supplied together with the program file (file extension .opd). When the program file is loaded on the destination system, BAPsplus will recognize and load the export library.

The assignment of program file and export library can be cancelled via 'Library/ Reload...'. The program file can then further be edited in the normal way. Commands that are only contained in the export library, are then no longer available. The initial assignment can be restored by selecting 'Library/Import...'.


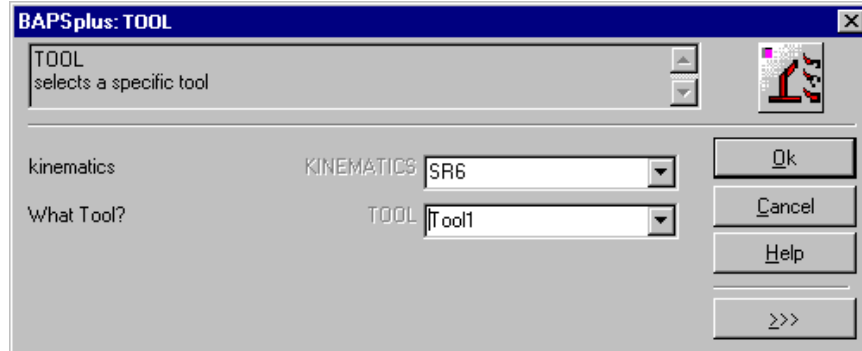
When an export library is loaded, the zoom window is no longer available. It will neither be possible to edit any libraries, the corresponding menu commands are deactivated.

Expanded Functions

5.9 Working with tools

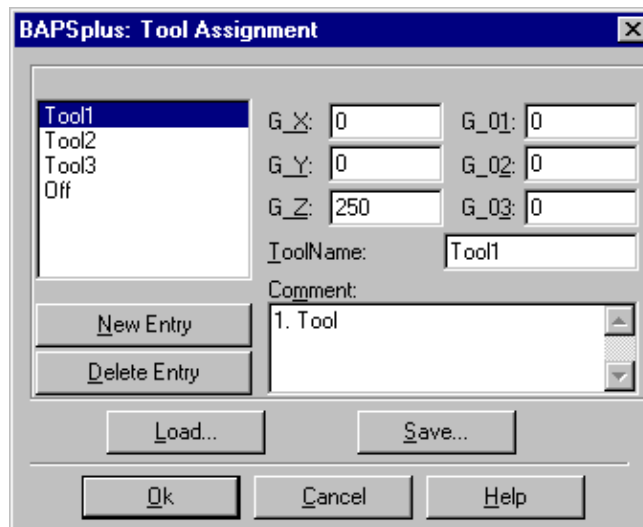
To be able to use different grippers or tools during the operation process, the BAPS programming language offers the tool command.

A new gripper can be selected in the BAPS program via its name:



The individual gripper parameters are defined in the dialog 'BAPSplus: Tool assignment' and saved in the file 'TOOLS.dat'.

- ★ Select 'Extras / Edit tools...' to open the dialog.



Three translations (G_X, G_Y, G_Z) and three rotations (G_D1, G_D2, G_D3) are used for the general description. They refer to the flange each.

- ☞ For further details about tools, see manual 'BAPS3 short description', section 'TOOLS.DAT'.

Data Input Manager

6 Data Input Manager

More and more end user ask for a desktop tailored their needs. On the one hand, they wish to make themselves easy modifications at the programs, on the other hand, they want to have to do as little as possible with the complex robot programs.

That is why functions for the generation of application specific programming surfaces have been integrated in BAPSplus. After having created the program, the actual BAPS source program is generated under the control of the Data Input Manager (DIM).

6.1 Function mode

Example:

The program contains a MOVE instruction moving from an intermediate position to an end position. The end user should be able to change the positions (declared at the beginning of the program) and the interpolation type.

First create the program framework:

- ★ create a new file.
- ★ insert a MOVE instruction.

Data Input Manager

- ★ In its parameter dialog no data, e.g. destination point, are entered. They are defined later.

With '>>>' the dialog is enlarged for further input possibilities. With 'Declarations...', the dialog for the variable declaration can be directly opened to declare variables, signals etc. subsequently.

In the list box 'Selection for DIM', the parameters are determined which can be changed in the following under DIM control.

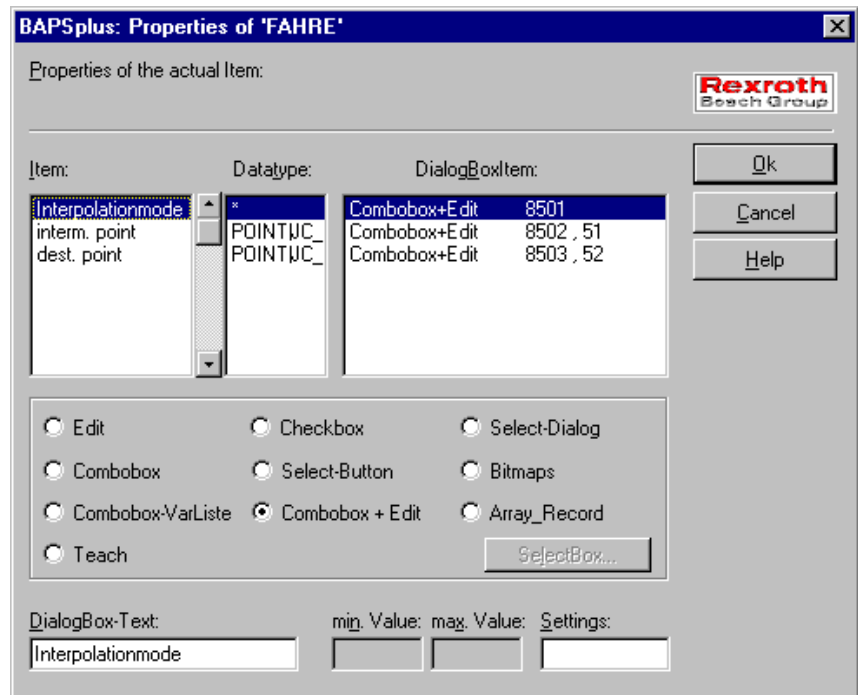
In the example, the following entries have been selected:

- Interpolation mode
- Intermediate point(s)
- destination point(s)

- ★ Confirm with 'Format...'

Data Input Manager

A dialog for determining the entry type of the selected parameters appears:



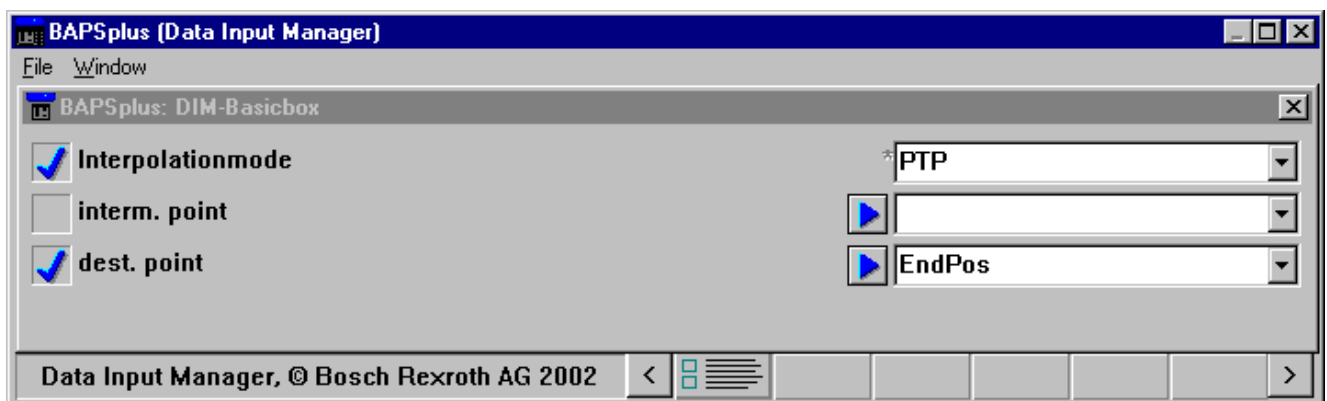
In the example, the following entry types have been defined:

- Edit field for interpolation type
- combination of combo box and variable list for intermediate point(s) and destination point(s).

☞ For later input, the corresponding data can be edited directly or selected from the list of the known variables.

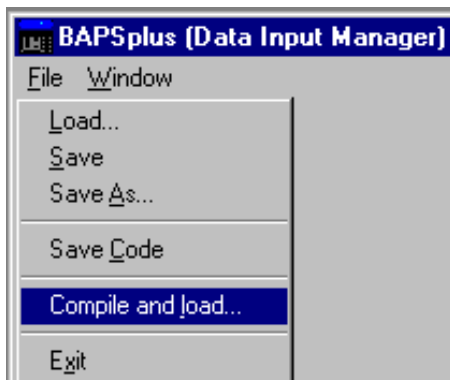
After closing all dialogs and saving the file, the data input manager can be activated via 'Extras/ Start DIM...'.

The data input manager hides the BAPSplus main window, loads the selected file and represents it in the following dialog:



Data Input Manager

The type of interpolation has been set to LINEAR. '@Viapunkt' has been selected as via point and '@dest_point' as target point.

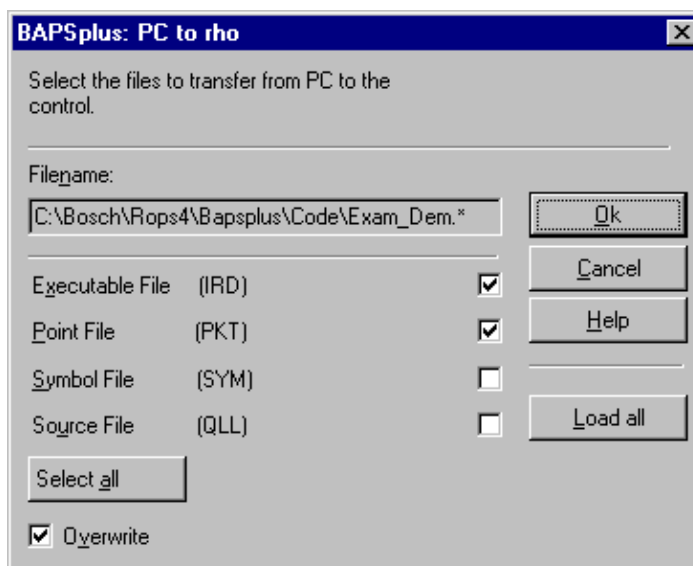
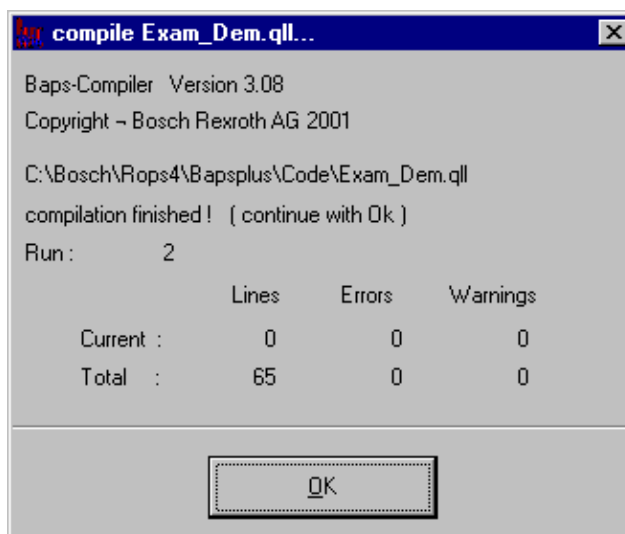


Activate 'File/Compile and load..' to generate the BAPS source program with the currently selected parameters:

```
;; WARNING -  
;; CONTROL = RHO4  
;; KINEMATICS: (1=SR800)  
;; SR800.WC_NAMES = X_K, Y_K, Z_K, A_K  
;; SR800.JC_NAMES = A_1, A_2, A_3, A_4  
  
PROGRAM BSP_DIM()  
  
    SR800.JC_POINT : @Via_point  
    SR800.JC_POINT : @dest_point  
  
BEGIN  
  
    ; Moving the robot from the current position  
    ; via an intermediate position into a target position  
    MOVE SR800 LINEAR VIA @Via_point TO @dest_point  
  
PROGRAM_END
```


Data Input Manager

This BAPS program is compiled and can be loaded into the control and executed provided that it is error-free:



Advantages for the end user:

- Only the parameters that have been marked by the programmer can be influenced when the DIM is in use.
- If the accessible parameters have been skilfully selected, wrong operation is very much reduced and the 'ready' program can still be adjusted to the current needs 'on site'.

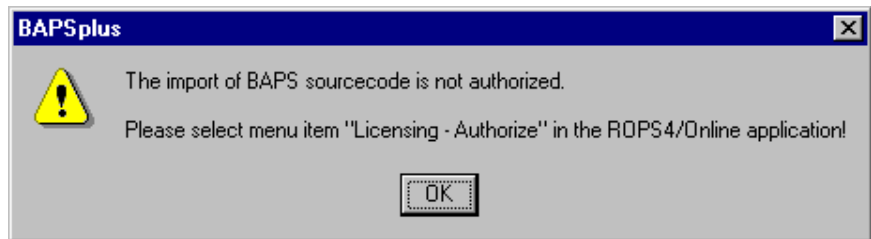
Data Input Manager

Notes:

7 BAPSplus Recompiler

The ReCompiler imports BAPS source texts (QLL files) and converts them into the BAPSplus file format (OPD file). The converted files are saved in the BAPSplus program file directory (in the standard installation the directory 'C:\Bosch\Rops4\BAPSplus\Dat') and can be edited further with BAPSplus. The file name of the resulting file corresponds to the BAPS source text file (file extension '.opd').

The use of the ReCompiler is subject to a license. If there is no valid license, the following dialog appears:



7.1 Function description

In this chapter are explained:

- the insertion of the ReCompiler into BAPSplus
- its functions
- and the marginal conditions that have to be taken into account when using the ReCompiler.



CAUTION

To ensure a correct import, the libraries (BIB files) listed in the following and delivered and installed with the graphic programming system BAPS-plus must not be modified.

The libraries are copied during the installation into the subdirectory '.\bibl' of the BAPSplus directory (in the standard installation 'C:\Bosch\Rops4\BAPSplus\Bibl):

- BAPS3.bib
- STANDARD.bib

These libraries must be loaded before calling the ReCompiler in BAPSplus.

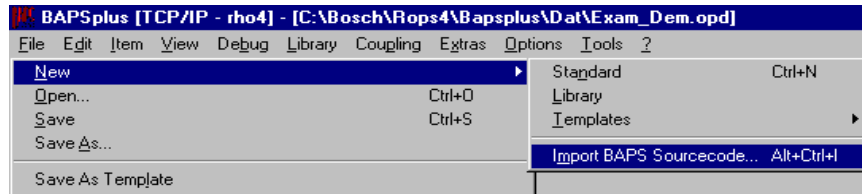
BAPSPplus Recompiler

7.2 Calling ReCompiler

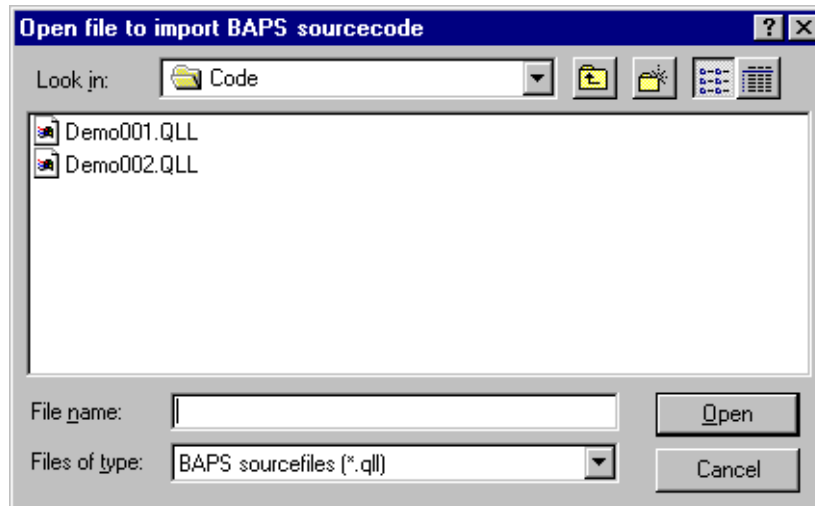
The ReCompiler can be called in different ways:


- item 'File/New/Import BAPS Sourcecode...'
- item 'File/Open ...'
- Drag & Drop from Windows Explorer

Item 'File/New/Import BAPS Sourcecode...'



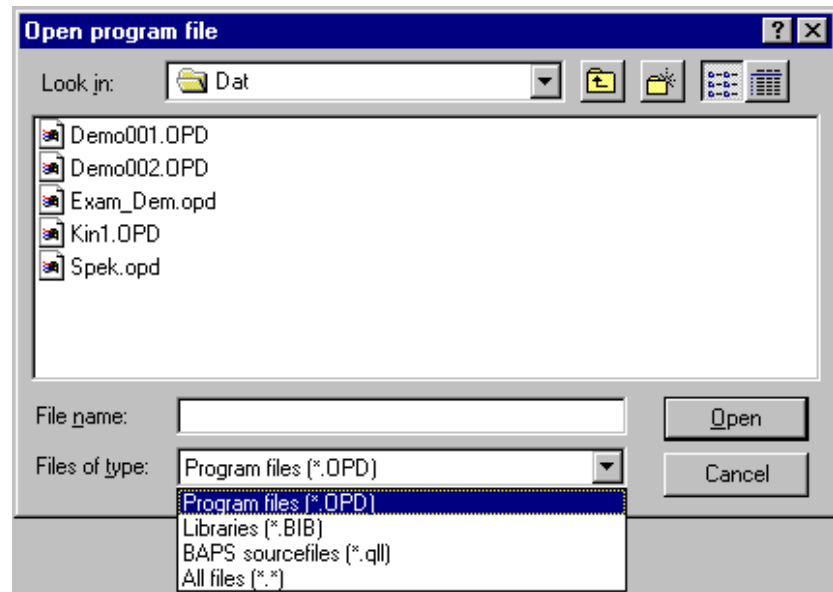
If the ReCompiler is licensed correctly, a dialog for selecting the BAPS source text to be imported appears:



 **Shortcut: <Ctrl+Alt+I>.**

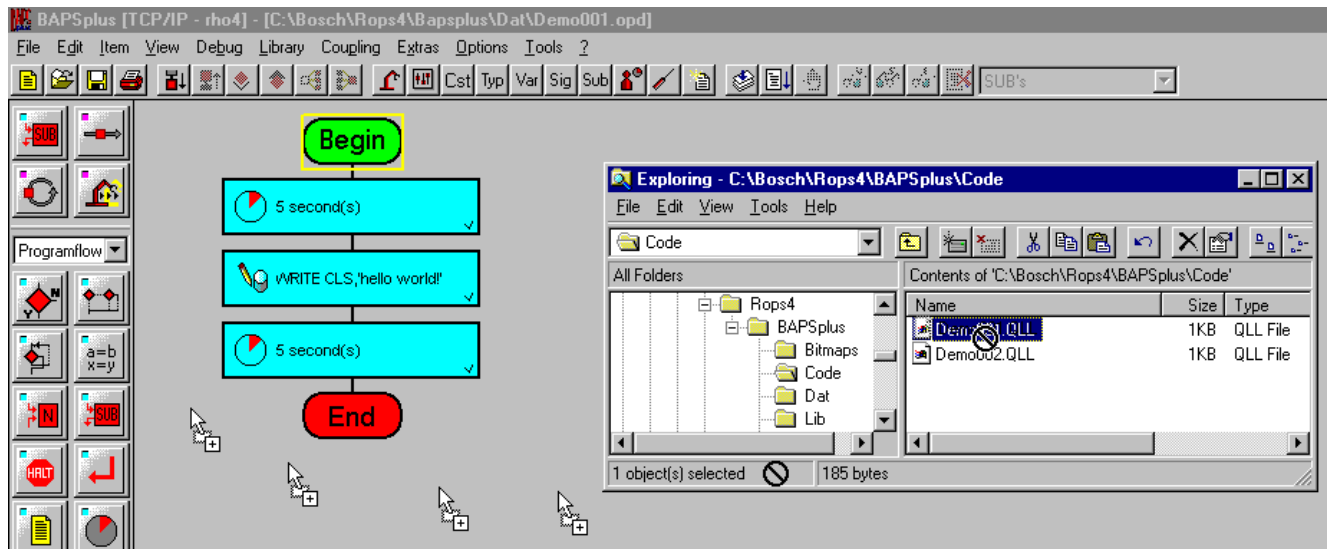
BAPSPplus Recompiler

Item 'File/Open...'



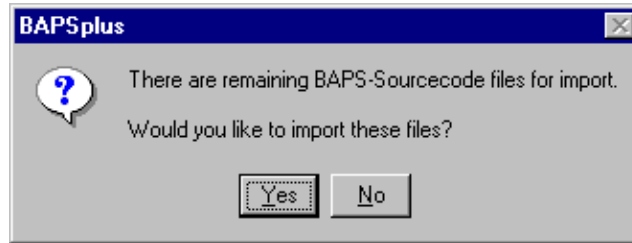
- ★ Set as file type 'BAPS sourcefiles (*.qll)'.

Drag & Drop from Windows Explorer



Several BAPS source texts can be processed in one time. If an irregularity arises, a message appears. The import of the remaining data can be ended.

BAPSplus Recompiler



Use Drag & Drop

- ★ Select in the Windows Explorer the BAPS source text file(s) to be imported.

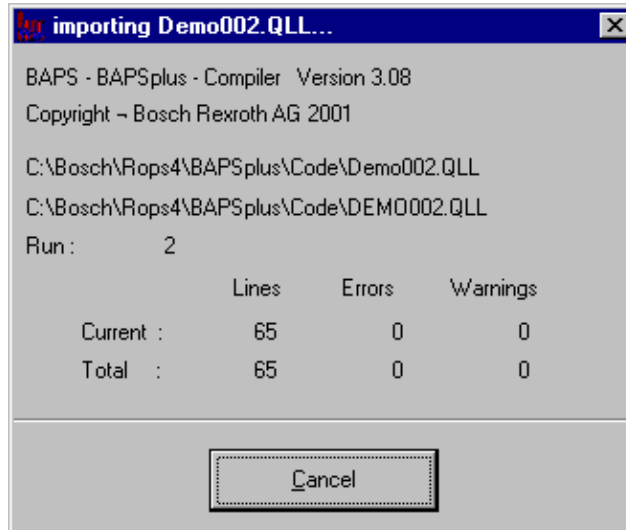
Ensure that BAPSplus is visible.

- ★ Transfer BAPS source text file(s) into the BAPSplus main window.

BAPSplus Recompiler

7.3 Import BAPS Sourcecode

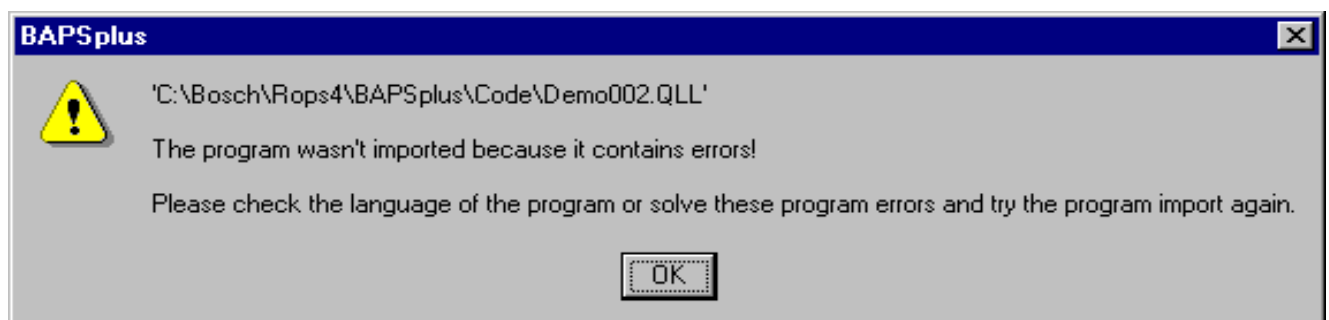
After selection of the desired file, the corresponding BAPS source text is imported. During the import, a corresponding information message appears:



The BAPSplus main window is locked for user entries. The program import in the status window of the ReCompiler and in the PFC can be followed at the same time.

- ☞ **The source text import can be interrupted at any time with 'Cancel', see page 7-3. The file that has been last edited will be loaded.**


If errors occur during the BAPS source text import, a dialog indicating the cause appears:



After confirming with 'OK', the active program before the program import will be loaded again.

The ReCompiler writes the program errors in the BAPSplus source text in an error file (file extension '.err') in the directory of the BAPS source text file. If there is no write access for this directory, the error file will be created in the temporary directory.

BAPSplus Recompiler

 **The name of the temporary directory appears in the entry 'TEMP' or 'TMP' in a DOS-window after entry of the command line 'SET'. If none of both entries are present, the main directory of the first hard disk (in general 'C:\') will be used as temporary directory.**

A point file of the imported BAPS source text is copied into the BAPSplus program code directory (in the standard installation 'C:\Bosch\Rops4\BAPSplus\Code'). The teach points that have been already defined are then available after the source text import.

If the entry 'CompileAfterImport' in the file 'BAPSplus.ini' is set (i.e. = 1), the program that has just been created after successful import will be compiled.

An OPD file with the same name in the BAPSplus program file directory will be saved. The file extension of this backup file is '.ibk'. If an IBK file of the same name exists, no further backup file will be created.

If the BAPS source text is imported from the BAPS plus program code directory and a BAPS source text file with the same name already exists, it will be saved with the file extension '.qbk'. If a QBK file already exists, no further backup file will be created.

The imported program BAPSplus is then available for further processing.

The ReCompiler protocols the source text import in a LOG file shown at the end, see 7.4.3. It contains information about the process of the source text import and issued status messages.

BAPSPplus Recompiler

7.4 Functions

The ReCompiler imports the following BAPS declarations and instructions:

Compiler instructions
Include, Error, Warning, Debuginfo, Ser_IO_Stop, File_Error, Process_Kind, Int, Control, Kinematics
Kinematic, Axis coordinate names
Declaration part
Constant declarations, type declarations, variable declarations, import declaration
Special function declarations
Rho function declarations
Program structure
Program header, parameter list, export declaration
Subroutine declarations, external main programs, return
Instruction part
Conditional instruction (IF .. THEN)
Branch instruction (CASE ..)
Repeat instruction (REPEAT ..)
Parallel instruction, exclusive instruction
Start instruction, Stop instruction
Move instruction, shift instruction
Record instruction, assignment, jump mark, jump instruction, SUBROUTINE call, Pause instruction, Halt instruction, Sync instruction, Synchron instruction, Synchron_end instruction, Wait instruction
Tool instruction, RefPnt instruction, limit_off instruction, assign instruction, close instruction, read instruction, write instruction, read_begin instruction, write_begin instruction, write_end instruction
Program slope, block slope
Standard functions
ORD, CHR, WC, JC, WC_Calc, Sizeof, Condition, End_of_file, SIN, COS, ATAN, Sqrt, Round, Trunc, ABS
Standard subroutines
Break, Command, Int_Asc_, Asc_Int, PLC_Process, PLC_Time

BAPSplus Recompiler

7.4.1 File name and file version of the BAPSplus program

The name of the generated OPD file corresponds to the name of the BAPS source text, '.opd' is used as file extension.

The version number of the resulting OPD file depends on the BAPSplus version from which the ReCompiler is called.

7.4.2 Language version of the BAPS source text

The language version of the BAPS source text to be imported corresponds to the language of the BAPSplus installation. It means that in the German version only German BAPS source texts and in the English version only English BAPS source texts can be imported.

7.4.3 Log file

The ReCompiler reports the source text import in a LOG file shown at the end (if not switched off), see 7.4.4. This protocol file contains information about the run of the source text import and issued status messages.


The log file is deposited in the temporary directory (e. g. C:\Windows\Temp). The file name is composed of the name of the BAPS source text file and the file extension '.log'.

The log file is displayed via the application registered with the file type '.log'. If no assignment exists, the standard Windows editor (Notepad.exe) is used.

BAPSplus Recompiler


Following entries can appear in the LOG file:

1. '<file>' This element, which should be imported, couldn't be found in the loaded libraries. The import of the selected BAPS sourcecode file isn't possible!

 **At least the libraries 'BAPS3' and 'STANDARD' must be loaded.**

- | | | |
|--------|---|---|
| Cause | : | The element <Name> is neither contained in the loaded libraries nor is a subroutine with the name <Name> in the BAPS source text or in the libraries defined. |
| Remedy | : | Insert corresponding library (item 'Library/Administrate...') or check the writing of the subroutine call. |

2. '<file>' The exporting BAPSplus file couldn't be opened.

 **For the BAPS sourcecode import of global data it is necessary that the exporting file must exist as a BAPSplus file. Please import the BAPS sourcecode of the exporting file first and try again the import of the importing file after that.**

- | | | |
|--------|---|--|
| Cause | : | The BAPSplus file to be exported (file extension '.opd') has not been found. It contains global data. |
| Remedy | : | Import first the BAPS source text of the file to be exported and save then the OPD file.

Check saving place of the BAPSplus file to be exported. It must be in the BAPSplus program file directory (see item 'Options/Directories...'). |

3. '<file>' This array consists of more than three dimensions. This isn't supported by BAPSplus!

- | | | |
|--------|---|---|
| Cause | : | A type of array or an array variable with more than 3 dimensions has been defined, e.g.:

<code>TArray4Dim = TArray4Dim = ARRAY[0..1] ARRAY[2..3] ARRAY[4..5] ARRAY[6..7] REAL</code> |
| Remedy | : | Group several dimensions together to one auxiliary type and use it as an element type, e.g:

<code>TArray1 = ARRAY[4..5] ARRAY[6..7] REAL
TArray4Dim = ARRAY[0..1] ARRAY[2..3] TArray1</code> |

BAPSPplus Recompiler

4. '<typename>' This type is saved as a library type.
- Cause : In one of the loaded libraries, a type of the same name is defined. This type of library replaces the type defined in the BAPS source text.
- Remedy : -
5. '<typename>' This type was newly generated and replaces the direct type declaration in line #. If this type leads to compiling errors later, you are able to rename it accordingly!
- Cause : The type of a structured variable has been defined directly during the variable declaration. The number of line indicates the line in the BAPS source text to be imported, e.g.:
- ```
RECORD
 INTEGER : G1
 REAL : D1
RECORD_END : VbndVar1
```
- Remedy : Define the type of the structured variable as an independent type and use it within the variable declaration, e.g.:
- ```
TYP: TRecord1 =RECORD
    INTEGER : G1
    REAL : D1
    RECORD_END
VAR: TRecord1 : VbndVar1
```
6. '<sub-name>' This subroutine is saved as a library subroutine.
- Cause : In one of the loaded libraries, a subroutine of the same name is defined. This library type replaces the subroutine defined in the BAPS source text.
- Remedy : -

BAPSplus Recompiler

7. The statement in the abort condition in line # may consist of only one simple statement. Please modify the QLL file accordingly and try the import again.

Cause : In a MOVE instruction, an abortion condition consisting of several instructions has been programmed, e.g.:

```
MOVE UNTIL E1=1 ERROR BEGIN WRITE 'What's now?' HALT END TO P1
```

Remedy : Re-program abortion condition, e.g.:

```
..
MOVE UNTIL E1=1 ERROR Message ('What's now?') TO P1
..
PROGRAMM_END
```

```
SUBROUTINE Message ( VALUE TEXT : Txt1 )
BEGIN
WRITE Txt1
HALT
SUBROUTINE_END
```

8. Sourceline: # The maximum number (max. 10) of parallel branches has already been reached. The remaining branches will be inserted into the last branch!

Cause : The maximum number of parallel branches is reached.

Remedy : Re-structure program.

9. Sourceline: # The maximum number (max. 10) of alternative branches has already been reached. The remaining branches will be inserted into the last branch!

Cause : The maximum number of alternatives is reached.

Remedy : Re-structure program.

10. The import of BAPS sourcecode was aborted by the user. The last edited file was reloaded!

Cause : The user has aborted the program import.

Remedy : -

11. <file> The PKT-file according to the imported BAPS sourcecode couldn't be copied to the BAPSplus programcode directory. Copy this file manually if you need the teach variables.

Cause : The point file corresponding to the imported BAPS source text could not be copied into the BAPSplus program code directory (point file already existing, access violation etc.)

Remedy : Check access rights on the file and copy the PKT file manually if required.

BAPSPplus Recompiler

12. '<file>' The program wasn't imported because it contains errors!
Please check the language of the program or solve these program errors and try the program import again.
Cause : There are faults in the BAPS program to be imported.
Remedy : Eliminate error in the BAPS source text.
13. <file> The BAPS sourcecode was successfully imported!
Cause : The import of the BAPS source text was successful.
Remedy : –
14. <file> The BAPS sourcecode was not imported!
The import of the BAPS source texts was not successful.
Remedy : Eliminate cause and retry import.

BAPSPplus ReCompiler

7.4.4 Settings of the ReCompiler

The working mode of the ReCompiler is influenced via entries in the file BAPSPplus.ini in the BAPSPplus program directory (in the standard installation 'C:\Bosch\Rops4\BAPSPplus').

Following entries are used. They can be found in the section [FileImport]:

Setting	Function
CreateLog	Determines if a LOG file is to be created (= 1, default) or not (= 0).
ShowLog	Indicates if the LOG file after the BAPS source text Import is to be displayed (= 1, default) or not (= 0).
UpdatePAP	If 'UpdatePAP = 1' (default), the PFC will be updated in continuous during the import. If the variable is not set (= 0), the PFC will be updated at the end (import more rapid).
AlwaysOverwritePKT	Determines if a point file that already exists is to be automatically overwritten (= 1) in the program code directory or not (= 0, default).
CompileAfterImport	Determines if after the successful BAPS source text import, the BAPS compiler is to be called and in this way the created program is to be compiled (= 1) or not (= 0, default).
MaxAssignLines	Determines the number of successive assignments that are saved in an assign element. The basic setting is 4. If other assignments follow, they will be saved in further assign elements.

BAPSplus Recompiler

Example for the settings of the ReCompiler in the file BAPSplus.ini

```
29.
30. ;-----
31. ; Diese Sektion enthält die Einstellungen für den Import
32. ; von BAPS Quelltexten!
33. ;
34. ; This section contains the settings for the import of
35. ; BAPS sourcecode.
36. ;
37. ; Syntax:
38. ;         CreateLog   = ( 0 | 1 )
39. ;         ShowLog     = ( 0 | 1 )
40. ;         UpdatePAP   = ( 0 | 1 )
41. ;
42. ; Beispiel: Die LOG-FILE soll angelegt und am END angezeigt werden,
43. ;           während des Imports wird der Programmablaufplan
gezeichnet:
44. ; Example: The LOG file should be created and shown after the import,
45. ;           the program flow chart is updated during import:
46. ;
47. ;         CreateLog=1
48. ;         ShowLog=1
49. ;         UpdatePAP=1
50. ;
51. [FileImport]
52.
53. CreateLog=1
54. ShowLog=1
55. UpdatePAP=1
56. AlwaysOverwritePKT=0
57. CompileAfterImport=0
58. MaxAssignLines=4
59.
```


7.5 Special cases



CAUTION

When using the ReCompiler, the following special cases must be taken into account.

Possible consequences, see example in 7.6

Compile instruction Warning

Compile instructions Warning (e. g. ';;Warning +') in the declaration part of the BAPS source text are grouped. The last one counts, i.e. it determines if warnings will be issued or not.

Compile instruction Kinematic declaration

If no kinematic is defined in the BAPS source text, the ReCompiler creates a kinematic (Name: New) with 2 axes (axis names: A1, A2, coordinate names: K1, K2).

Compile instruction Kinematic change

Multiple conversions of the default kinematic (e.g. ';;Kinematics = SR6') in the declaration part are grouped. The last one determines which kinematic is the default kinematic at the beginning of the instruction part.

Compile instruction Control

Multiple conversions of the target control (e. g. 'Control = rho4') appears in the declaration part are grouped. The last one determines the valid target control.

Compile instruction Interpolation

When this compile instruction (e.g. ';;INT = Linear') appears in the declaration part, it is shifted into the application part of the program or subroutine.

Compile instruction Error

When this compile instruction (e. g. ';;ERROR = 10') appears several times in the declaration part, only the last one is interpreted.

Compile instruction Debuginfo

When this compile instruction (e. g. ';;DEBUGINFO +') appears several times in the declaration part, only the last one is interpreted.

Compile instruction Ser_IO_Stop

When this compile instruction (e.g. ';;SER_IO_STOP -') appears several times in the declaration part, only the last one is interpreted.

Compile instruction Fileerror

When this compile instruction (e.g. ';;FILE_ERROR +') appears several times in the declaration part, so only the last one is interpreted.

Compile instruction Include

Include instructions (e.g. ';;INCLUDE VarDek1.inc'). The source text of the file is directly imported.

BAPSplus Recompiler

Import of array types and variables

Array types with a maximum of 3 dimensions are supported. If more are required, see 7.4.3, Number 3 on page 7–9.

Import of implied type and variable declarations

Implied type definitions are expanded. This means, e.g. 'INPUT : 1 = Inp1' becomes 'INPUT BINARY: 1 = Inp1'.

Import of MOVE instructions with cancel condition

MOVE instructions with cancel condition consisting of one instruction can be imported. If more instructions are required, see 7.4.3, Number 7 on page 7–11.

Variable initializations

Assignments at the beginning of the program are not converted into variable initializations.

If a BAPS source text created with BAPSplus is recompiled, the initializations contained in it will be inserted as assignments at the beginning of the program or subroutine.

This means, the regenerated OPD file is not identical with the original file, the BAPS source texts and the running time behavior of the programs are however the same.

Import of global data

For the correct import of BAPS source texts importing the global variables, the following conditions are to be met, see 7.4.3, Number 2 on page 7–9.

- The file exporting global variables must be compiled. This means the SYM and PKT files must be available in the directory of the BAPS source text to be imported (PKT file only if teach points exist).
- The file exporting global variables must exist as OPD file. It means it must be imported as BAPS source text **before** the file importing these global variables.

Import of parallel instructions

Parallel instructions with a maximum of 10 parallel branches are imported. If more branches are programmed, the BAPS source text import is interrupted.

Import of branch instructions (CASE instructions)

Branch instructions with a maximum of 10 alternatives are imported. If more branches are programmed, the remaining branches will be included into the DEFAULT branch.

Comments in the BAPS source texts

Comments contained in BAPS source texts are **not** taken into account during the import. If necessary, they must be included manually into the generated BAPSplus program (OPD file).

 **Alternative: Open source file in an editor. Copy comments and insert in corresponding entry fields.**

BAPSPplus Recompiler

BAPSPplus Program library(ies)

Import of BAPS source texts (QLL files) and saving as BAPSPplus program library is not possible.

BAPSPplus Recompiler

7.6 Example: BAPS source text import

- BAPS source text of the file to be imported (contains include file)
- created log file
- source text of the imported file

BAPS source text of the file to be imported

```

60.
61. ; Includefile VarDecl1.inc
62.
63.   NAMSTRING : Name
64.   INPUT     : 1 = In1
65.   POINT     : P1
66.

1.   ;;ERROR = 10
2.   ;;DEBUGINFO -
3.   ;;Ser_IO_Stop -
4.   ;;File_Error -
5.   ;;Warning -
6.   ;;Warning +
7.
8.   PROGRAM Demo
9.
10.  ;;INT = LINEAR
11.
12.  CONST:
13.    LLimit = 0,
14.    ULimit = 1
15.
16.  TYPE:
17.    TArrayCnst = ARRAY[LLimit .. ULimit] CHAR
18.    TArrayCalc = ARRAY[LLimit .. LLimit+10] CHAR
19.    TArrayDim = ARRAY[0..1] ARRAY[2..3] ARRAY[4..5] ARRAY[6..7] REAL
20.    TArray1 = ARRAY[4..5] ARRAY[6..7] REAL
21.    T4Dim = ARRAY[0..1] ARRAY[2..3] TArray1
22.    TRecord1 = RECORD
23.        INTEGER : I1, I2, I3, I4, I5, I6, I7, I8, I9
24.        ; comment
25.        ; INTEGER : I10, I11
26.        REAL : R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11
27.    RECORD_END
28.    TRecInt = RECORD
29.        INTEGER : I1, I2, I3, I4, I5, I6, I7, I8, I9, I10, I11
30.    RECORD_END
31.    TRecord2 = RECORD
32.        TRecInt : RecInt
33.        REAL : R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11
34.    RECORD_END
35.    NAMSTRING = ARRAY [ 1 .. 20 ] CHAR
36.

```

BAPSpplus Recompiler

```
37.  VAR:
38.    RECORD
39.      INTEGER : I1
40.      REAL    : R1
41.    RECORD_END : RecVar1
42.
43.    RECORD
44.      INTEGER : I1, I2, I3, I4, I5, I6, I7, I8, I9, I10, I11
45.    RECORD_END : RecVar2
46.
47.    TArray4Dim : V1
48.    T4Dim      : V2
49.    TRecord1   : V3
50.    TRecord2   : V4
51.
52.  ;;INCLUDE VarDecl1.INC
53.
54.  BEGIN
55.    RecVar1.I1 = RecVar2.I3
56.    Name = 'Name'
57.    V1[0][2][4][6] = 11.11
58.    V2[0][2][4][6] = 22.22
59.    V3.I9 = V3.I1
60.    V3.R11 = V3.R1
61.    V4.RecInt.I9 = V4.RecInt.I1
62.    V4.R11 = V4.R1
63.    MOVE TO In1=1 ERROR BEGIN WRITE 'What now?' HALT END TO P1
64.    MOVE TO In1=1 ERROR Message ('What now?') TO P1
65.  PROGRAM_END
66.
67.  SUBROUTINE Message ( VALUE TEXT : Txt1 )
68.  BEGIN
69.    WRITE Txt1
70.    HALT
71.  SUBROUTINE_END
```

BAPSPplus Recompiler

Log file

BAPSPplus LOG-File: BAPS source text Import

BAPS source file: **D:\BAPSPplus\Code\Demo.qll**

TArray4Dim

This field has more than 3 dimensions.

This is not directly supported by BAPSPplus and can lead to a 'somewhat complicated' further use of this type.

NAMSTRING

This data type is saved as library type.

T_AutoTyp01

This type has been re-generated and replaces the direct type declaration in line 38.0.

If this type leads to errors during the compilation, it can be renamed.

T_AutoTyp02

This type has been re-generated and replaces the direct type declaration in line 43.0.

If this type leads to errors during the compilation, it can be renamed.

The error instruction in line 63.0 must only consist of one simple instruction.

Please re-edit the QLL source correspondingly and try again to import.

D:\BAPSPplus\Code\Demo.qll

The BAPS source text has been successfully imported.

BAPSpplus Recompiler

BAPS source text of the imported file

```

72.  ;;ERROR = 10
73.  ;;DEBUGINFO -
74.  ;;SER_IO_STOP -
75.  ;;FILE_ERROR -
76.  ;;KINEMATICS: (1=New)
77.  ;;New.WC_NAMES = K1, K2
78.  ;;New.JC_NAMES = A1, A2
79.  PROGRAM Demo
80.
81.  CONST:
82.    LLimit = 0,
83.    ULimit = 1
84.
85.  TYPE:
86.    TArrayCnst = ARRAY[0..1] CHAR
87.    TArrayCalc = ARRAY[0..10] CHAR
88.    TArray4Dim = ARRAY[0..1] ARRAY[2..3] ARRAY[4..5] ARRAY[6..7] REAL
89.    TArray1 = ARRAY[4..5] ARRAY[6..7] REAL
90.    T4Dim = ARRAY[0..1] ARRAY[2..3] TArray1
91.    TRecord1 = RECORD
92.      INTEGER : I1, I2, I3, I4, I5, I6, I7, I8, I9
93.      REAL : R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11
94.    RECORD_END
95.    TRecInt = RECORD
96.      INTEGER : I1, I2, I3, I4, I5, I6, I7, I8, I9, I10, I11
97.    RECORD_END
98.    TRecord2 = RECORD
99.      TRecInt : RecInt
100.     REAL : R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11
101.    RECORD_END
102.    NAMSTRING = ARRAY[1..12] CHAR ; library type (see rmain.bib) !
103.    T_AutoTyp01 = RECORD
104.      INTEGER : I1
105.      REAL : R1
106.    RECORD_END
107.    T_AutoTyp02 = RECORD
108.      INTEGER : I1, I2, I3, I4, I5, I6, I7, I8, I9, I10, I11
109.    RECORD_END
110.
111.  VAR:
112.    T_AutoTyp01 : RecVar1
113.    T_AutoTyp02 : RecVar2
114.    TArray4Dim : V1
115.    T4Dim : V2
116.    TRecord1 : V3
117.    TRecord2 : V4
118.    NAMSTRING : Name
119.    POINT : P1
120.    INPUT BINARY : 1 = In1
121.
122.  BEGIN
123.    ;; INT = LINEAR

```

BAPsplus Recompiler

```
124.  RecVar1. I1 = RecVar2. I3
125.  Name = 'Name'
126.  V1[ 0][ 2][ 4][ 6] = 11.11
127.  V2[ 0][ 2][ 4][ 6] = 22.22
128.  V3. I9 = V3. I1
129.  V3. R11 = V3. R1
130.  V4. RecInt. I9 = V4. RecInt. I1
131.  V4. R11 = V4. R1
132. MOVE LINEAR UNTIL In1=1 *only single instruction allowed TO P1
133. MOVE LINEAR UNTIL In1=1 ERROR Message('What now?') TO P1
134. PROGRAM_END
135.
136. SUBROUTINE Message (VALUE TEXT: Txt1)
137.
138. BEGIN
139.   WRITE Txt1
140.   HALT
141.
142. SUBROUTINE_END
```

* Recompiler error message (see chapter "7.4.3 Log file", point 7)

Setting possibilities of the INI file

8 Setting possibilities of the INI file

The file 'BAPSplus.ini' allows to change the BAPSplus behavior and look. Every user can adapt BAPSplus to his needs.

 **The corresponding standard value for a parameter is given in brackets.**

8.1 TOOLS configuration

The section [TOOLS] allows to modify the menu entry 'Tools'. For detailed information, see chapter 4.2.10.

```
[Tools]
```

```
&Editor=C:\Bosch\Rops4\Edit.exe
```

```
&Online=C:\Bosch\Rops4\Online.exe
```

```
SEPARATOR=
```

```
&Commandline=C:\WINNT\System32\CMD.EXE
```

Setting possibilities of the INI file

8.2 Import configuration

The behavior of BAPSplus during the QLL file import can be configured via the following parameters in section [FILEIMPORT]:

CreateLog=(0 | [1])

Creates a logfile for the corresponding import and saves it in the set Temp directory.

ShowLog = (0 | [1])

Shows the import log file at the end of the import.

UpdatePAP=(0 | [1])

To be able to follow the creation of the PFC during the import, this setting must be activated. The import is accelerated when this entry is set on '0', i.e. the current updating is deactivated.

AlwaysOverwritePKT=([0] | 1)

Defines if a point file that is already existing in the program code directory is to be automatically overwritten (= 1) or not (= 0, default setting).

 **In the default setting, an inquiry appears to ask if the point file can be overwritten.**

CompileAfterImport=([0] | 1)

If this entry is active (=1), the program is directly saved and compiled after the successful import.

 **Activate this entry when via Drag & Drop several files in succession are to be imported. No inquiry comes to ask if the individual files are to be saved. They are automatically saved.**

Example:

- Create LOG file
- show it at the end
- draw PFC during import

```
[FileImport]
```

```
CreateLog=1
```

```
ShowLog=1
```

```
UpdatePAP=1
```

Setting possibilities of the INI file

8.3 BAPSPplus setting

Basic configurations of the development environment are set in section [BAPSPplus] of the file 'BAPSPplus.ini'.

Standard fill color

The standard fill color for program elements is 0x00FFFF00, the color format is composed of the parts red(r), green(g) and blue(b).

```
DefFillColor=0x00bbggrr
```

Standard dimensions for program elements


The following values are the standard values for these settings. There is no absolute minimum or maximum value:

```
MaxSizeX=200
```

```
MinSizeX=80
```

```
MaxSizeY=100
```

```
MinSizeY=40
```

 **Recommendation:**
Width ≤ half screen resolution,
Height ≤ 2 x standard value (MinSizeY).

```
GotoLastPos= 0 | [ 1 ]
```

Allows to position the cursor at the place last edited of the program to be loaded to go on working at the same place where the program was left.

Language setting

The BAPSPplus language can be changed.

```
Language=( [Deutsch] | English )
```

 **Compiler must be available in the corresponding language.**

If no entry has been found in 'BAPSPplus.ini', the setting is taken from the group [Install] of the file 'ROPS4WIN.ini'.

 **Do not change the setting**

Setting possibilities of the INI file

SplashScreen

This entry enables to set the display time of the SplashScreen (in seconds). '0' suppresses the SplashScreen.

```
Delay=1
```

Undo

The maximum number of UNDO steps can be defined freely and is only limited through the free place on the hard disk. The number of the REDO steps is coupled to UNDO (Default value 90).

```
MaxUndo=90
```

```
ActiveOnBreak= ( 0 | [ 1 ] )
```


Via this entry, it is possible to set the reaction of BAPsplus to the reached break points.

In the default setting, BAPsplus is taken automatically in the foreground as soon as a break point is reached. This is also the case when BAPsplus runs in the background and is not visible. Via the entry '0' this behavior is deactivated.

Size of internal source text buffer

Modifies the maximum size of internal buffers for the representation of the source text window.

```
MaxBufferSize=[0x10000] (corresponds to 64KB)
```

 **If BAPsplus issues the message that the memory for the representation of the source text is not sufficient, this value should be increased and BAPsplus restarted.**

Signal sound

This option allows to switch on or off the beep sound when messages are issued.

```
DoMessageBeep=0 | [ 1 ]
```

Automatic name suggestion

In the declaration of types, variables, signals etc., BAPsplus creates automatically corresponding name suggestions. Via the following settings, this automatism can be adapted. BAPsplus uses then the preset prefix to create the suggestion name.

Setting possibilities of the INI file

```
CompPrefix= [m_]
ParPrefix= [P]
VarPrefix= []
SigPrefix= []
TypPrefix= [T]
SUBPrefix= [Up]
```

For components (`CompPrefix`), a name is suggested on the basis of type and prefix (e. g.: `m_Real1`).

In a similar way, this also applies to parameters (`ParPrefix`) and type designations (`TypPrefix`).

Variables (`VarPrefix`), signals (`SigPrefix`) and subroutines (`SUBPrefix`) have no prefix as standard definition.

Only the first both signs of the prefix are used. Exception: prefix `SUBPrefix`, for which the first three signs are adopted.

Template directory

This entry allows to adapt the template directory to individual needs. If nothing is entered there, the template files are searched in the directory 'Templates' in the BAPSplus directory (in the standard installation 'C:\Bosch\Rops4\BAPSplus\Template').

```
TemplateDir= [Template]
```

Assign lines

The maximum number of lines that are represented graphically within an assign element.

This value is limited through the maximum height of the corresponding PFC element:

```
MaxAssignLines= [4]
```

Quickinfo

These options allow to switch on or off the quickinfos. The delay time (in milliseconds) going by until the info window displayed after the last mouse motion can be also set.

```
QuickInfo= [ 1 ] | 0
QuickInfoDelay= [500]
```

Setting possibilities of the INI file

8.4 Settings of the incremental compiler

In the group [ShowCompErrors], individual error messages and warnings can be deactivated. This setting for warnings and errors have no effect on the normal compiler. This means that even the messages that are here hidden are displayed by the external BAPS compiler.

```
[ShowCompErrors]
```

```
6217=0 ; warning! def.JC_NAMES or WC_NAMES used  
1486=0 ; prog. and file name differ  
6215=0 ; error count exceeded  
6247=0 ; warning! implicit declaration of a point
```

8.5 Other groups

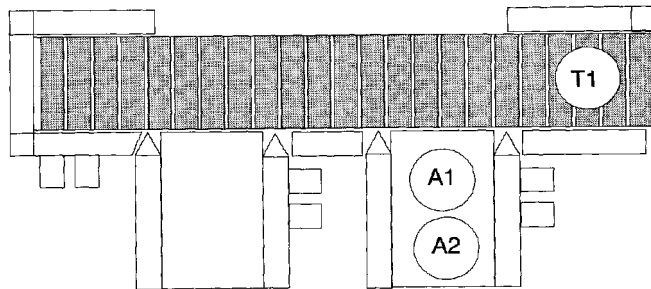
BAPSplus also uses a few other groups for internal information.

 **They are written dynamically and should not be modified.**

Example

9 Example

A few advantages of BAPSplus are shown here on the basis of a simple example.



Task of the 'turboscara' SR6:

Place pieces A1 and A2 on the place T1. Since the position T1 is occupied after the first depositing by A1, the belt must convey A1 away.

Basic configuration and elementary commands of BAPSplus

The basis for each program is the declaration of constants, types, variables and signals. They enable a simple and clear access to the used data.

- ☞ **Declare the variables with names 'speaking for themselves' to facilitate the reading of the program. Up to 12 characters can be used for names.**

BAPSplus lists these names at the required places automatically.

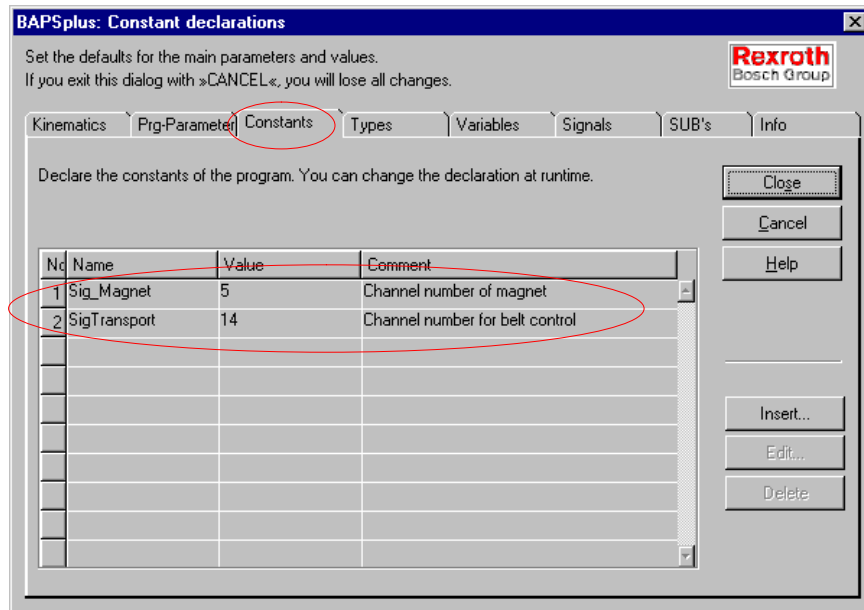
Define constants

Two constants must be defined as channel numbers for binary outputs:

- **Sig_Magnet**
The constant switches on or off the magnet and activates the transportation.

Example

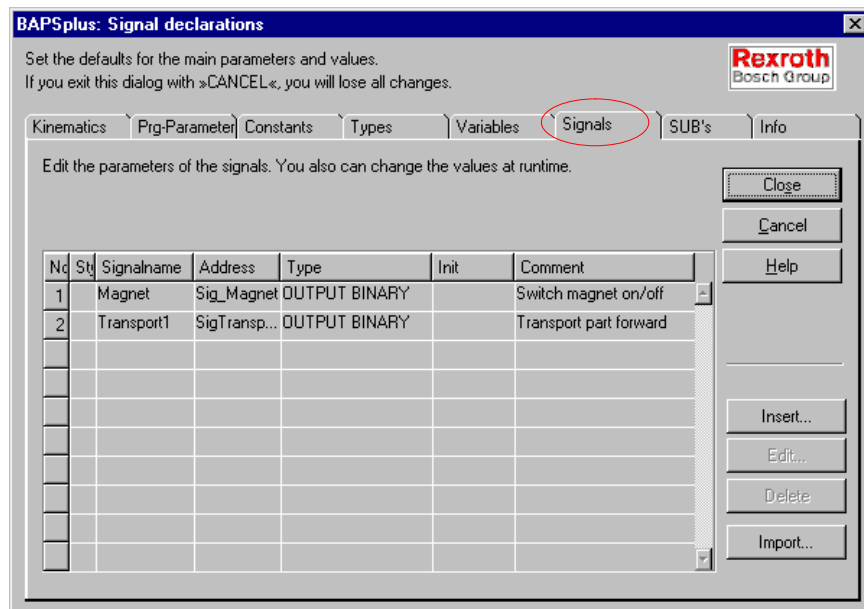
- Sig_Transport
 'Transport' means that the transportation will be started. 'Sig' is used as abbreviation for 'Signal'.



Define signals

Two other binary outputs are required to:

- switch on or off the magnet
- activate the transportation



Example

Declare variables

In this example, the following items are required:

- A1Below
- A1Above
- T1Below
- T1Above

These points are taught and therefore characterized as DEF variables.

Example of declaration dialog for the variable 'A1below':

BAPSplus: Declare variable

Set the parameter of the variable.
Init value and comment are optional!

Datatype: POINT

Name: A1Below

Kinematics: SR6

DEF-Variable
 Exportable

Import file: _____


Init value: _____

Comment: Position A1 below

The configuration part for Step01 is then completed. For further information about the declaration, see corresponding chapter.

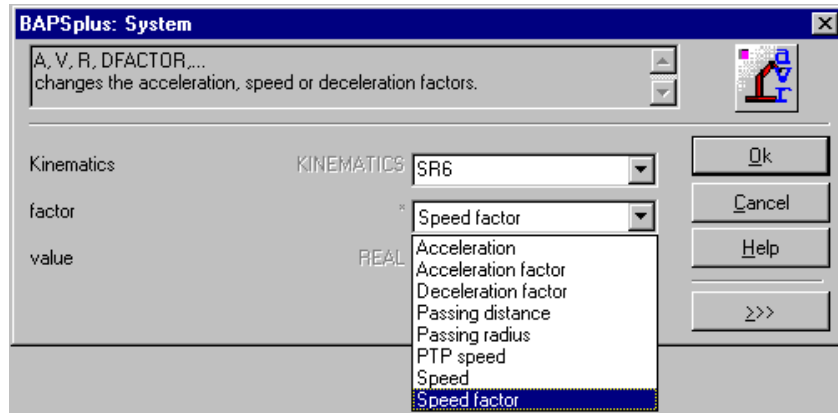
Insert movement instructions

- AFACTOR (acceleration factor), VFACTOR (velocity factor), DFAC-
TOR (deceleration factor)
- BLOCK_SLOPE

 **For detailed information, see manual 'BAPS3 Programming manual', section 'Block transitions'.**

Example

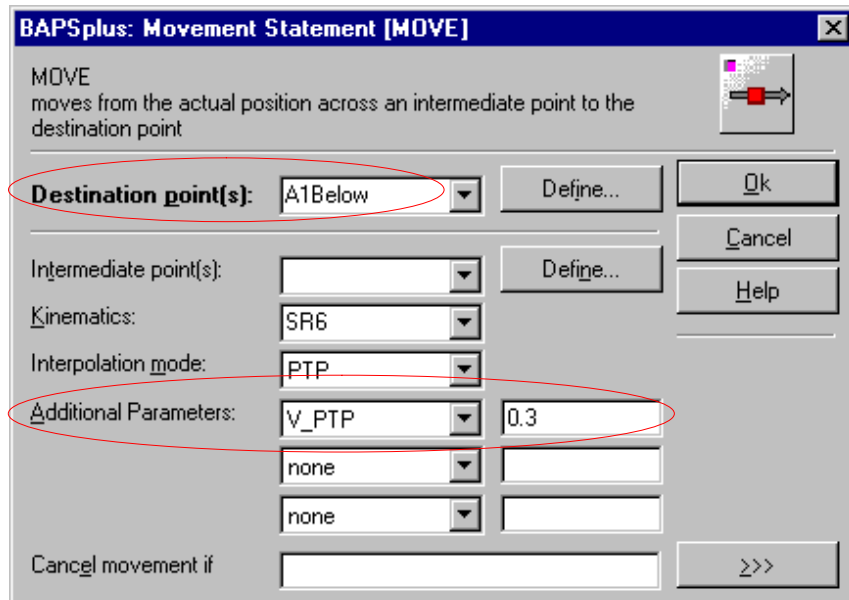
The first block is used for the setting of 'Factors'. In this case, the factor is only used to set the VFACTOR (velocity factor) on '1'. The block slope is inserted a the corresponding place.



The VFACTOR is suitable for testing since the velocity of the program can be reduced with a modification. The program slope enables to make a motion more fluent since a block slope after each block first reduces the velocity down to 0.

MOVE instruction

A MOVE instruction is next required to move the robot to the initial position, defined as 'A1Above'.



Example

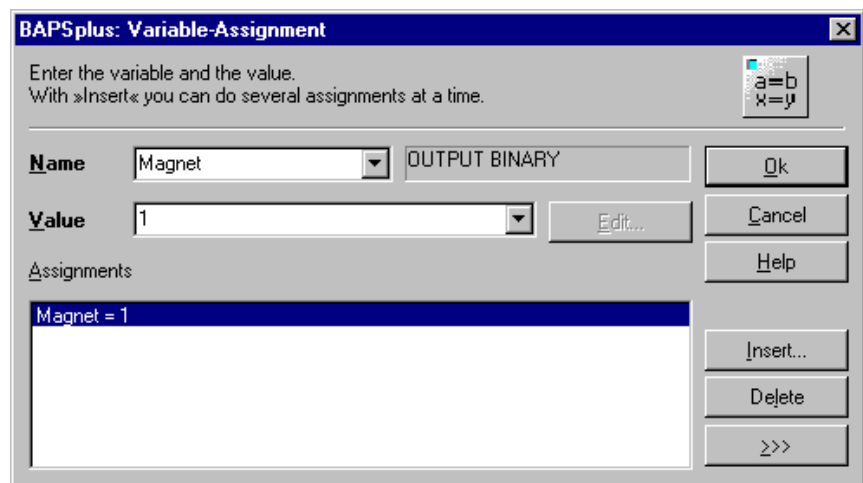
Since the movement is to be performed as quickly as possible, it is carried out in PTP (point-to-point) mode. This means that all axes start running at the same time and reach the final point at the same time. The movement itself does not need to be linear.

The movement to the piece 'A1Below' to be picked up should be a straight line. Additionally, the velocity will be reduced to 30% for running down.

- ☞ **With the VFACTOR, the global value for the velocity is set. Additionally, it is possible to define a velocity locally (only for the current movement). It must be taken into account that only circle and straight line have a velocity. Since PTP has no constant velocity, it is possible to program a time instead of a velocity.**

For the piece to be picked up, the magnet must be switched on via the instruction 'Magnet = 1'.

- ☞ **Remark: Acknowledge entry with 'OK'.**

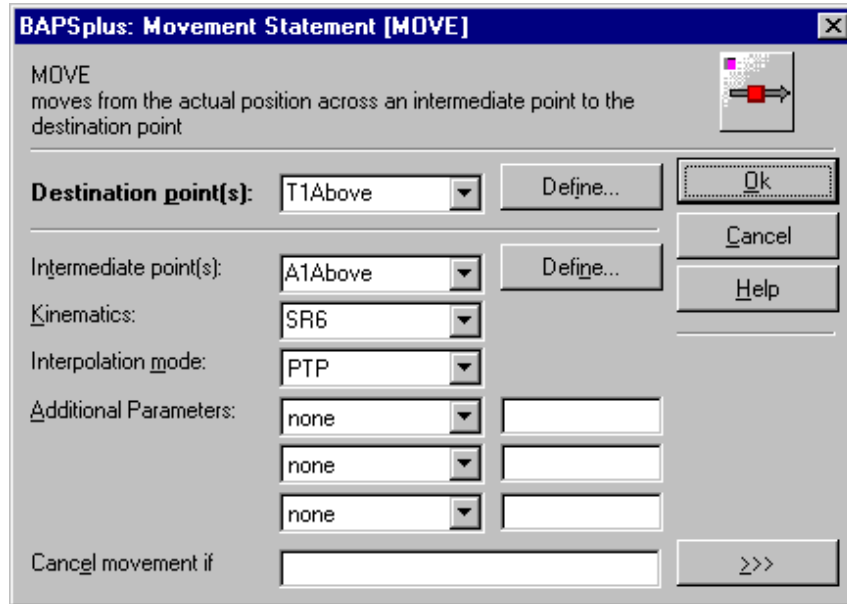


MOVE via intermediate point

The next movement goes up to position 'T1Above'. It can be performed more quickly since it need not be an exact straight line. The part A1 must be lifted on a straight line because a collision could be otherwise possible.

Example

One can think at first that the movement will be divided into two MOVE instructions. Inconvenient: Two points must be negotiated exactly. For this reason, the two movements are brought together to one movement. The position 'T1Above' cannot be directly negotiated, that is why an intermediate point, 'A1Above' is indicated.



After the position over the deposit position has been reached, the movement must be programmed downward. For this case the same condition applies as for picking up. This means that the movement is performed only with 30% of the velocity. At the end, the magnet is switched off with the instruction: Magnet = 0. Move then the gripper with the movement type PTP into the position 'T1Above'.

For the transportation of the part, another signal must be created at the output 'Transport'. Since the model used here requires a positive slope, the input must be shortly switched off and then on. This is realized through two instructions and a short waiting period (approx. 0.3 second).

The program with BAPSplus is now created.

Example

PFC and resulting BAPS source text from Step01

The screenshot displays the BAPSplus software interface for Step01. The main window shows a ladder logic program with the following steps:

- Begin
- SR6 . VFACTOR = 1.0
- ;;KINEMATICS = SR6
BLOCK_SLOPE
- MOVE SR6 PTP TO A1↓
- MOVE SR6 PTP WITH \
- Magnet = 1
- MOVE SR6 PTP VIA A:
- MOVE SR6 PTP WITH \
- Magnet = 0
- MOVE SR6 PTP TO T1↓
- Transport = 0
- 0.3 second(s)
- Transport = 1
- End

The right-hand pane shows the BAPS Sourcecode:

```

1 ;;KINEMATICS: (1=SR6)
2 ;;SR6.WC_NAMES = X_A, Y_A, Z_A, C_A
3 ;;SR6.JC_NAMES = A_1, A_2, A_3, A_4
4 PROGRAM step01
5
6 CONST:
7   Sig_Magnet = 5,
8   SigTransport = 14
9 VAR:
10  SR6.POINT : A1Below
11  SR6.POINT : A1Above
12  SR6.POINT : T1Above
13  SR6.POINT : T1Below
14  OUTPUT_BINARY : Sig_Magnet = Magnet
15  OUTPUT_BINARY : SigTransport = Transport
16
17 BEGIN
18  SR6 . VFACTOR = 1.0
19  ;;KINEMATICS = SR6
20  BLOCK_SLOPE
21  MOVE SR6 PTP TO A1Above
22  MOVE SR6 PTP WITH V_PTP = 0.3 TO A1Below
23  Magnet = 1
24  MOVE SR6 PTP VIA A1Above TO T1Above
25  MOVE SR6 PTP WITH V_PTP = 0.3 TO T1Below
26  Magnet = 0
27  MOVE SR6 PTP TO T1Above
28  Transport = 0
29  WAIT 0.3
30  Transport = 1
31 PROGRAM_END
32

```

The status bar at the bottom indicates: Michael Kaczorowski (BRC/... Main program La: 1 [0] Ln: 30 Value assignment(s) to variable(s)

Example














Source text

```
1.   ;;KINEMATICS: (1=SR6)
2.   ;;SR6.WC_NAMES = X_A, Y_A, Z_A, C_A
3.   ;;SR6.JC_NAMES = A_1, A_2, A_3, A_4
4.   PROGRAM Step01
5.
6.   CONST:
7.     Sig_Magnet = 5,
8.     SigTransport = 14
9.   VAR:
10.  DEF SR6.POINT : A1Below
11.  DEF SR6.POINT : A1Above
12.  DEF SR6.POINT : T1Above
13.  DEF SR6.POINT : T1Below
14.  DEF SR6.POINT : A2Above
15.  DEF SR6.POINT : A2Below
16.  OUTPUT BINARY : Sig_Magnet = Magnet
17.  OUTPUT BINARY : SigTransport = Transport
18.
19.  BEGIN
20.    SR6.VFACTOR = 1.0
21.    ;;KINEMATICS = SR6
22.    BLOCK_SLOPE
23.    MOVE SR6 PTP TO A1Above
24.    MOVE SR6 PTP WITH V_PTP = 0.3 TO A1Below
25.    Magnet = 1
26.    MOVE SR6 PTP VIA A1Above TO T1Above
27.    MOVE SR6 PTP WITH V_PTP = 0.3 TO T1Below
28.    Magnet = 0
29.    MOVE SR6 PTP TO T1Above
30.    Transport = 0
31.    WAIT 0.3
32.    Transport = 1
33.  PROGRAM_END
```









Shortcuts and tool bars

10 Shortcuts and tool bars

The following shortcuts and tool bars are available within BAPSplus:

Shortcut	Symbol	Signification
–		Select signals for observation
–		Select variables for observation
–		Select variables of external processes for observation
–		Open dialog 'Kinematics declarations'
–		Open dialog 'Constant declarations'
–		Open dialog 'Type declaration'
–		Open Open dialog 'Variable declarations'
–		Open dialog 'Signal declarations'
–		Open dialog 'Subroutine declarations'
–		Open dialog 'Display edit list'
–		Open dialog 'Tool assignment'
–		Open dialog 'Edit command'
Alt+A	–	Rearrange all levels
Alt+C	–	Open or close code display
Alt+E	–	Rearrange current level
Alt+Ins	–	Insert CASE alternative
Alt+Enter	–	Open property dialog of the active PFC symbol
Alt+Del	–	Delete CASE alternative
Alt+F	–	Activate icon overview
Alt+F3	–	Start search
Alt+F8		End test mode
Alt+F9	–	Step Out
Alt+H	–	Hierarchy
Alt+Ctrl+S	–	Save BAPS source text
Alt+Z	–	Zoom window

Shortcuts and tool bars

Shortcut	Symbol	Signification
Backspace		Change into the higher level when a lower level in PFC is re-presented.
Cursor to the left	-	To the left element
Cursor to the top		To the previous element
Cursor to the right	-	To the right element
Cursor to the bottom		To the next element
Enter	-	Change into a lower level when the active element in PFC is a block element.
del	-	Delete element
F3		Continue search
F4	-	Copy from rho to PC
F5	-	Copy from PC to rho
F6		Compile current source file
F7	-	Go to cursor
F8	-	Select debug mode
F9	-	Step Into
F11	-	Complete screen
F12	-	Icon overview
Ctrl+A	-	Mark all
Ctrl+B		Pause or continue
Ctrl+C	-	Copy mark in clipboard
Ctrl+F8		Start program
Ctrl+G		Group element
Ctrl+F	-	Start search
Ctrl+N		Create a new file with the standard settings
Ctrl+O		Open existing file
Ctrl+P		Print file
Ctrl+S		Save current file
Ctrl+U		Undo groups

Shortcuts and tool bars

Shortcut	Symbol	Signification
Ctrl+V	–	Paste from clipboard
Ctrl+W	–	Cut mark
Ctrl+X	–	Re-draw
Ctrl+Y	–	Redo
Ctrl+Z	–	Undo
Shift+Enter	–	Open parameter dialog of the active PFC symbol
Shift+F10	–	Open context menu
Shift+F8	–	Continue program execution

Shortcuts and tool bars

Notes:

Appendix

A Appendix

A.1 Abbreviations

Abbreviation	Meaning
ASCII	American Standard Code for Information Interchange
BAPS3	Programming language; Bewegungs- und Ablaufprogrammiersprache, Version 3
C:	Hard disk drive C
DDE	Dynamic Data Exchange
DDEML	Dynamic Data Exchange Management Library
EGB	Elektrostatic sensitive components
ESD	Electrostatic discharge
JC	Machine coordinates
MPP	Machine parameter program
PC	Personal Computer
PE	Protective Earth
POS	Actual position
PG	Programming unit
RC	Robot Control
RCO	Robot Control Output
ROPS4	Robot programming system for rho4
TCP/IP	Transmission Control Protocol/Internet Protocol
OC	Original coordinates
WC	World coordinates

Appendix

A.2 Index**A**

array type, 5–8

B

BAPS source code, 3–20

block generation, 2–1, 3–6

C

code window, 3–20

context menu, 3–5

D

Data Input Manager, 6–1

Data type, 3–3

debugging, 2–1

DIM, 2–2, 6–1

Documentation, 1–7

Drag&Drop, 7–3, 7–4

E

EMC Directive, 1–1

EMERGENCY–STOP devices, 1–5

ESD

Electrostatic discharge, 1–6

grounding, 1–6

workplace, 1–6

ESD-sensitive components, 1–6

F

file types

configuration files, 3–4

default kinematics file, 3–4

default tool files, 3–4

error files, 3–4

export library files, 3–3

initialization files, 3–4

library files, 3–3

point files, 3–4

program files, 3–3

rho4 program files, 3–4

source files, 3–4

symbol files, 3–4

tool file, 3–4

first steps, 3–5

Floppy disk drive, 1–7

full screen, 4–19

G

Grounding bracelet, 1–6

H

Hard disk drive, 1–7

header, 3–8

hierarchy, 3–22

I

icon, 4–18

background, 5–23

coloration, 5–25

group symbol, 5–24

icon bar, 3–16

Incremental Compiler, 8–6

L

level display, 3–22

Low-Voltage Directive, 1–1

M

menu bar, 3–8

Modules sensitive to electrostatic discharge. See
ESD-sensitive components**O**

overview display, 3–21

P

PFC, 3–19

program flowchart, 3–19

programming section, 3–19

Q

Qualified personnel, 1–2

R

Recompiler, 7–1

log file, 7–18

special cases, 7–15

Status window, 7–5

Recompiler log file, 7–8

Recompiler settings, 7–13

record type, 5–9

Release, 1–8

remote debugging, 2–1

requirements, 2–2

additional software, 2–2

hardware, 2–2

operating system, 2–2

ROPS4, 3–1

S

Safety instructions, 1–4

Safety markings, 1–3

source code, 3–20

Spare parts, 1–6

Appendix

Standard operation, 1-1
status line, 3-20
symbol, 2-1, 3-16

T

templates, 5-21
Test activities, 1-5
tool bar, 3-15
top-down methodology, 2-1, 3-6
Trademarks, 1-8

V

variable declaration, 5-10

Z

Zoom window, 3-21, 4-18

Appendix

Notes:

Bosch Rexroth AG
Electric Drives and Controls
P.O. Box 13 57
97803 Lohr, Germany
Bgm.-Dr.-Nebel-Str. 2
97816 Lohr, Germany
Phone +49 (0)93 52-40-50 60
Fax +49 (0)93 52-40-49 41
service.svc@boschrexroth.de
www.boschrexroth.com

