

SIEMENS

SINUMERIK 840D/810D

Description of Functions

09.2001 Edition

Configuring the OP 030 Operator Interface

SIEMENS

SINUMERIK 840D/810D

Configuring the OP 030 Operator Interface

Description of Functions

Valid for

<i>Control</i>	<i>Software Version</i>
SINUMERIK 840D	6
SINUMERIK 840DE (Export version)	6
SINUMERIK 840D powerline	6
SINUMERIK 840DE powerline (Export version)	6
SINUMERIK 810D	3
SINUMERIK 810DE (Export version)	3
SINUMERIK 810D powerline	6
SINUMERIK 810DE powerline (Export version)	6

09.01 Edition

Operator's Guide	BA
Development Kit	EU
Screen Kit	IK
Introduction to Configuring	PSE
References	A

SINUMERIK® Documentation

Printing history

Brief details of this edition and previous editions are listed below.

The status of each edition is shown by the code in the "Remarks" column.

Status code in the "Remarks" column:

- A** New documentation.
- B** Unrevised reprint with new Order No.
- C** Revised edition with new status.
If factual changes have been made since the last edition, this is indicated by a new edition coding in the header.

Edition	Order No.	Remarks
02.95	6FC5 297-2AC40-0BP0	A
04.95	6FC5 297-2AC40-0BP1	C
09.95	6FC5 297-3AC40-0BP0	C
09.01	6FC5 297-6AC40-0BP0	C

This book is included with the documentation on CD-ROM (DOCONCD)

Edition	Order No.	Remarks
01.02	6FC5 298-6CA00-0BG2	C

Trademarks

SIMATIC®, SIMATIC HMI®, SIMATIC NET®, SIROTEC®, SINUMERIK® and SIMODRIVE® are registered trademarks of Siemens AG. Some other designations used in these documents are also registered trademarks; the owner's rights may be violated if they are used by third parties for their own purposes.

You will find further information on the Internet at:
<http://www.ad.siemens.de/sinumerik>

This publication was produced with WinWord V8.0, Designer V7.0 and the AutWinDoc documentation tool.

The reproduction, transmission or use of this document or its contents is not permitted without express written authority. Offenders will be liable for damages. All rights, including rights created by patent grant or registration of a utility model or design, are reserved.

Other functions not described in this documentation might be executable in the control. This does not, however, represent an obligation to supply such functions with a new control or when servicing.

We have checked that the contents of this document correspond to the hardware and software described. Nonetheless, difference might exist and we cannot therefore guarantee that they are completely identical. The information contained in this document is, however, reviewed regularly and any necessary changes will be included in the next edition. We welcome suggestions for improvement.

Preface

Organization of the documentation

SINUMERIK documentation is arranged in three parts:

- General documentation
- User documentation
- Manufacturer/service documentation

SINUMERIK 840D powerline

The improved-performance

- SINUMERIK 840D powerline and
- SINUMERIK 840DE powerline

have been available since 09/2001. For a list of available **powerline** modules, please refer to the Hardware Description /PHD/ in Section 1.1.

SINUMERIK 810D powerline

The improved-performance

- SINUMERIK 810D powerline and
- SINUMERIK 810DE powerline

have been available since 12/2001. For a list of available **powerline** modules, please refer to the Hardware Description /PHC/ in Section 1.1.

This Manual is intended for use by



- Programmers
- Service and operating personnel

Standard scope

The manual consists of the following sections:

- /BA/ Operator's Guide
- /EU/ Development Environment (Configuring Package)
- /IK/ Screen Kit: Software Update and Configuration
- /PSE/ Introduction to Configuring the Operator Interface

/PS/ Online only: Configuring syntax (configuring package)
This document is supplied with the software as a pdf file.

Search assistance	To ease your orientation, in addition to the main table of contents, we have provided an index and table of contents by chapter for each manual. You will also find a list of literature references in the appendix.
Qualified personnel	A device may only be commissioned and operated by qualified personnel. Qualified personnel as referred to in the safety guidelines in this document are persons who are authorized to start up, earth and label devices, systems and circuits in accordance with the relevant safety standards.
Proper use	Please note the following:
	<hr/> <p>Warning</p> <p>The device may only be used for the applications described in the catalog and in the technical description, and only in combination with equipment, devices and components from other manufacturers approved or recommended by Siemens.</p> <p>The safe and fault-free functioning of the device can only be guaranteed if its transport, storage, assembly and installation, as well as its operation and maintenance have been carried out correctly.</p> <hr/>
	<hr/> <p>Note</p> <p>This indicates a piece of important information relating to the product and/or its operation, or highlights part of the documentation which requires special attention.</p> <hr/>
Safety guidelines	This manual contains information intended to ensure your personal safety as well as to prevent damage to products. Safety notices are highlighted by a warning triangle and categorized according to level of risk as follows:
Notes	The following notes and symbols used as markers in the documentation are particularly important:
	<hr/> <p>Note</p> <p>This symbol indicates that additional information follows.</p> <hr/>
	<hr/> <p>Important</p> <p>This symbol indicates that important information follows which must be observed.</p> <hr/>



Additional ordering options

Appears in the documentation whenever a described function is not contained in the standard version but may be ordered as an option.

Warnings

The manual contains the following warnings that indicate various levels of danger:



Danger

Indicates an imminently hazardous situation which, if not avoided, **will** result in death or serious injury or in substantial property damage.



Warning

Indicates a potentially hazardous situation which, if not avoided, **could** result in death or serious injury or in substantial property damage.



Caution

Used with the safety alert symbol indicates a potentially hazardous situation which, if not avoided, **may** result in minor or moderate injury or in property damage.

Caution

Used without safety alert symbol indicates a potentially hazardous situation which, if not avoided, **may** result in property damage.

Notice

Used without the safety alert symbol indicates a potential situation which, if not avoided, **may** result in an undesirable result or state.

Technical information

Trademarks

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



SINUMERIK 840D/810D

Configuring the OP 030 Operator Interface

Operator's Guide (BA)

Introduction	BA/1-3
1.1 Requirement	BA/1-4
1.2 Use.....	BA/1-5
1.3 Screen Brightness.....	BA/1-5
Control Panel	BA/2-7
2.1 Operator panel.....	BA/2-8
2.2 Key actions	BA/2-10
2.3 Graphical user interface.....	BA/2-11
2.3.1 Screen partitioning, softkeys.....	BA/2-11
2.3.2 Operating principle.....	BA/2-13
2.3.3 Menu trees.....	BA/2-14
Operation	BA/3-19
3.1 User areas – Main display	BA/3-20
3.2 Actual value display	BA/3-21
3.3 Current block display	BA/3-22
3.4 Program control	BA/3-22
3.4.1 Program control – Single block.....	BA/3-23
3.5 Input and display of R parameters.....	BA/3-24
3.6 Work offsets: Overview.....	BA/3-24
3.7 Work offsets: Translation, rotation.....	BA/3-25
3.8 Work offsets: Scale and mirror	BA/3-25
3.9 Tool overview.....	BA/3-26
3.10 Tool offsets	BA/3-28
3.11 Tool management.....	BA/3-28
3.11.1 Load tool	BA/3-30
3.11.2 Unload tool.....	BA/3-31
3.11.3 Display and modify tool data	BA/3-32
3.11.4 Find empty location.....	BA/3-33
3.11.5 Select location in current magazine	BA/3-34
3.11.6 Select tool list to load	BA/3-34
3.11.7 Create new tool.....	BA/3-35
3.12 Workpiece overview.....	BA/3-36
3.13 Overview of the global main programs	BA/3-37

3.14	Overview of the global subroutines.....	BA/3-37
3.15	Display window	BA/3-38
3.16	Program selection.....	BA/3-38
3.17	Alarm and message overview.....	BA/3-39
3.18	System.....	BA/3-39
3.19	Read in data.....	BA/3-40
Index	BA/4-45

1

Introduction

1.1	Requirement	BA/1-4
1.2	Use.....	BA/1-5
1.3	Screen Brightness.....	BA/1-5

1.1 Requirement

Control The OP 030 must be connected to a SINUMERIK 840D/810D via MPI. The software version of the SINUMERIK 840D/810D must be approved for the OP 030.

OEM-specific extensions OEM-specific extensions, such as PLC alarms and message texts and any dedicated user interface components, must be installed on the OP 030 for operation with a machine tool.

Software update The software on the OP 030 can be updated with the OP 030 screen kit. You will find a description of the software updating process in the following document:

Reference: /FBO/, IK, Screen Kit

Attention

The software update should only be performed by suitably trained personnel.

1.2 Use

Target group	This documentation is intended for end users of the SINUMERIK 840D/810D control with the OP 030 operator panel on a machine tool.
Objective	<p>This Operator's Guide is intended to provide the end user with a brief overview of the basic operating functions.</p> <p>This Operator's Guide also contains a description of the operating sequences defined by SIEMENS as standard user areas.</p>

1.3 Screen Brightness

Runup	After Power ON or Reset the OP 030 runs up, displays the version and tries to make contact with the NC for approx. 90 seconds. If this is not successful, it must be acknowledged before a new attempt can be made.
Brightness setting	During the runup period the "+" and "-" keys can be used to set the brightness of the screen display. "+" increases the brightness. Unfortunately, this setting is not accepted in the executable HMI program.



2

Control Panel

2.1	Operator panel	BA/2-8
2.2	Key actions	BA/2-10
2.3	Graphical user interface.....	BA/2-11
2.3.1	Screen partitioning, softkeys.....	BA/2-11
2.3.2	Operating principle.....	BA/2-13
2.3.3	Menu trees	BA/2-14

2.1 Operator panel

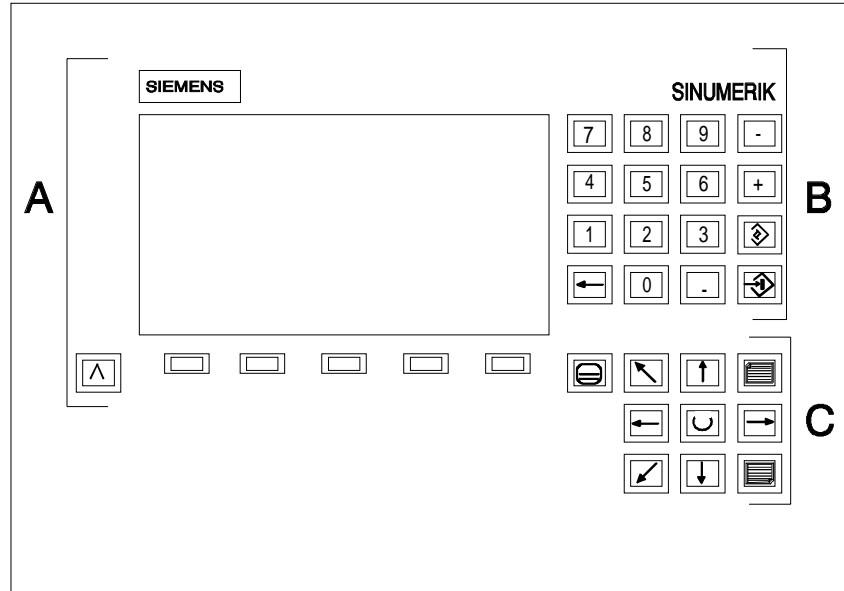


Fig. 2-1 Operator panel OP 030 with 14cm monitor (screen diagonal), resolution: 240x128 pixels

- A** Graphics monitor, monochrome, 5 horizontal softkeys
- B** Numeric keypad with Override and Input key
- C** Cursor pad



Recall key: recalls the previous window.



Area Switchover key: displays the main menu with the user areas.

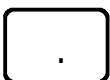
The selectable areas/functions are printed on the softkeys. When the Area Switchover key is pressed twice in succession, the previously selected menu reappears.



Digits 0 to 9



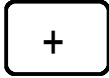
Backspace key, deletes characters from right to left.



Period



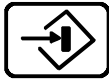
Minus, subtraction



Plus, addition



Edit key: switches from navigation mode to edit mode (insert) in tables and input boxes.
Reset – discards the input when pressed repeatedly (undo).



Input key: saves an edited value



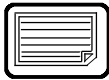
Cursor Up, Down, Right, Left



Home (cursor to beginning of document or screen page)

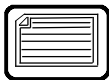


End (cursor to end of document or screen page)



Page Down

When a scroll bar is visible in the window, you can display more data with the Page Up/Down keys.



Page Up



Selection key

Selection key for default values in fields identified by this key symbol.

2.2 Key actions

Navigation mode	In this mode, you can switch between the individual fields with the cursor control keys and the Page Up and Page Down keys.
Edit mode: overwrite field contents	Position the cursor on the field you want to overwrite. Enter the digits in the field. Before you enter the digits the contents of the field are deleted automatically by the system and the field is empty.
Edit mode: change field contents	Position the cursor on the field you want to overwrite. Open the field with the Edit key. The contents of the field are retained. You can now use the arrow keys to position the cursor in the field in order to edit the contents.
Save the input	You can save the input using the Input key or by editing the field with the Up Arrow or Down Arrow keys.
Undo	If you press the Edit key again in edit mode, the data entered in your field are not saved. The field is closed and you are returned to navigation mode. The value entered in the field before you switched to edit mode reappears in the field.
Delete one character	In edit mode, you can use the Backspace key to delete the character to the immediate left of the cursor.
Calculator functions: addition, substraction	Depending on their configuration, the fields are equipped with a calculator function enabling addition and subtraction operations to be performed on the field contents. Open the field in edit mode for insertion with the Edit key. Enter + or –. Enter your second operator (value). Complete the entry with the Input key. The result of the calculation appears in the field.

2.3 Graphical user interface

2.3.1 Screen partitioning, softkeys

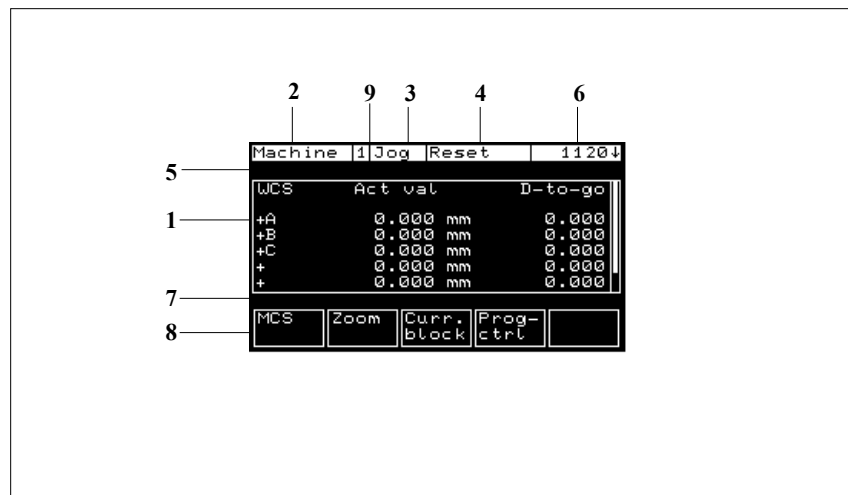


Fig. 2-2 Screen regions

- 1 **Working window** and NC displays
- 2 **User area display**
Machine, parameter, program, services, diagnosis
- 3 **Operating mode display:** Jog, Mda, Auto
- 4 **Channel status display**
 - Channel reset
 - Channel active
 - Channel interrupted

- 5 Channel message line**
 1. Stop: No NC ready
 2. Stop: No mode group ready
 3. Stop: EMERGENCY STOP active
 4. Stop: Alarm with stop active
 5. Stop: M0/M1 active
 6. Stop: Block in single block terminated
 7. Stop: NC stop active
 8. Wait: Read-in enable missing
 9. Wait: Feed enable missing
 10. Wait: Dwell time active
 11. Wait: Aux. fct. acknowledgement missing (Aux. fct. = auxiliary function)
 12. Wait: Axis enable missing
 13. Wait: Exact stop not reached
 14. Wait: For positioning axis
 15. Wait: For spindle
 16. Wait: For other channel
 17. Wait: Feed override at 0%
 18. Stop: NC block error
 19. Wait: For external NC block
 20. Wait due to SYNACT instruction
 21. Wait: Block search active

- 6 Alarm code display:** display of the alarm number of the alarm which occurred last. The symbol ↓ appears after the alarm number if several alarms exist.

- 7 Dialog line:** user prompting

- 8 Softkey bar** with 5 softkeys.

- 9 Channel number**

2.3.2 Operating principle

The main menu branches into 7 user areas:

1. Machine
2. Parameter
3. Program
4. Alarm
5. System (version, language)
6. Config. (interface for application programming)

The main level is selected using the Area Switchover key



5 softkeys

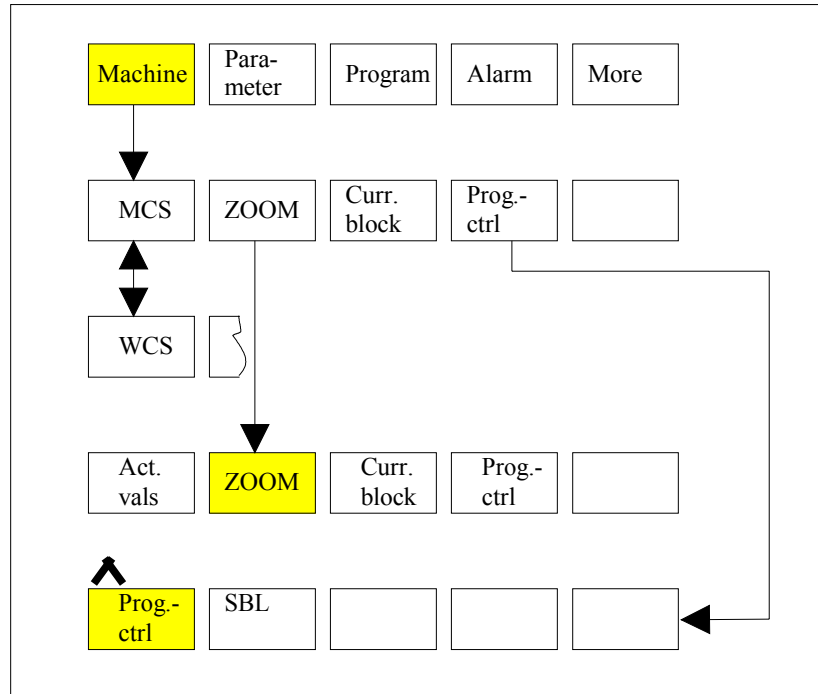
The softkeys divide a user area into subareas or branch to functions.



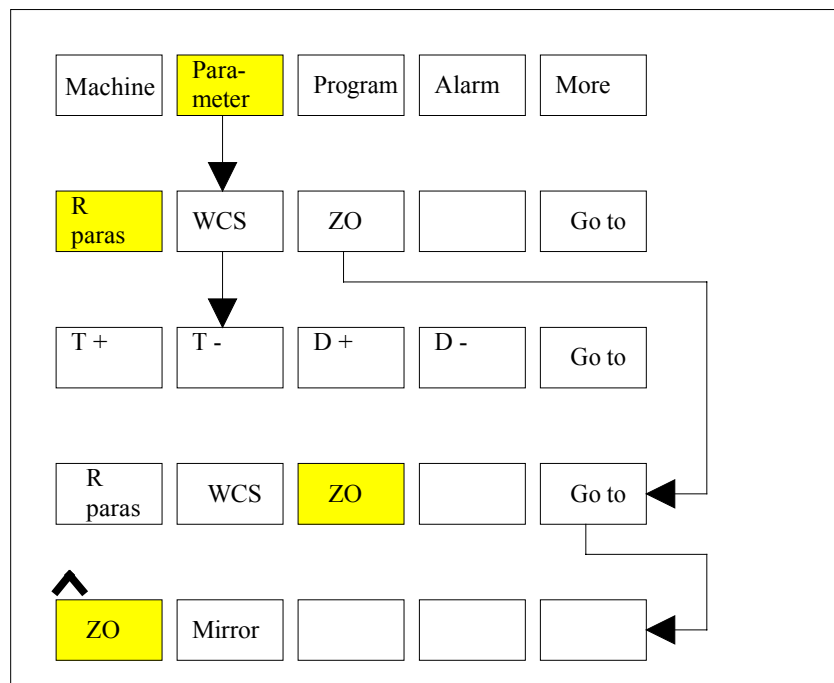
Recalls the previous window.

2.3.3 Menu trees

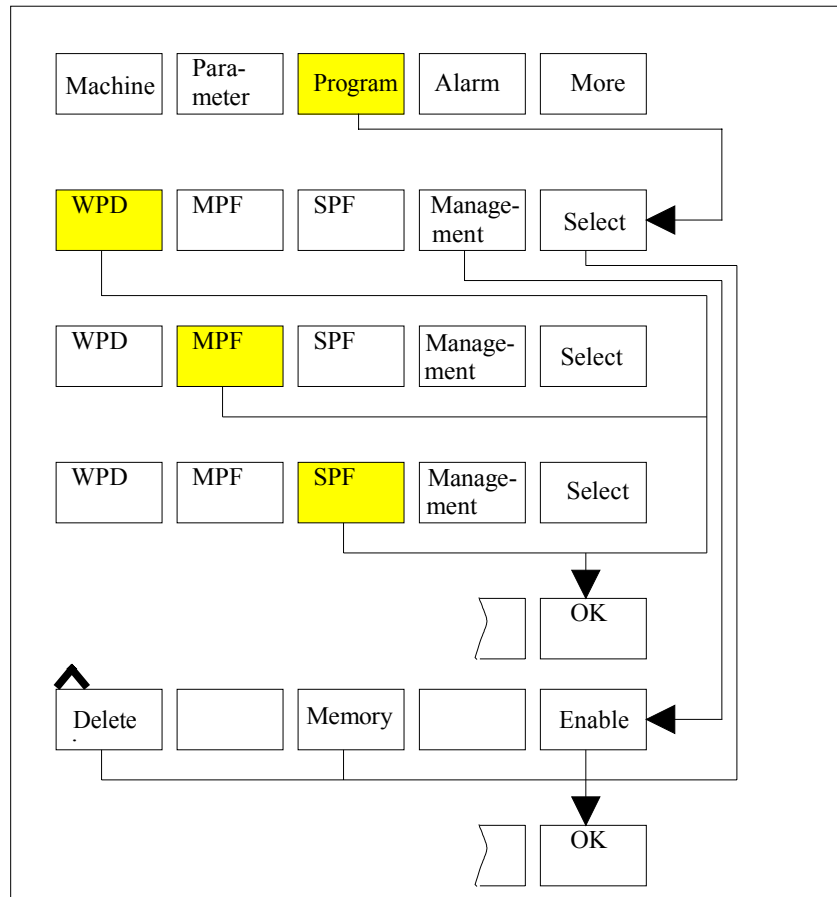
Machine



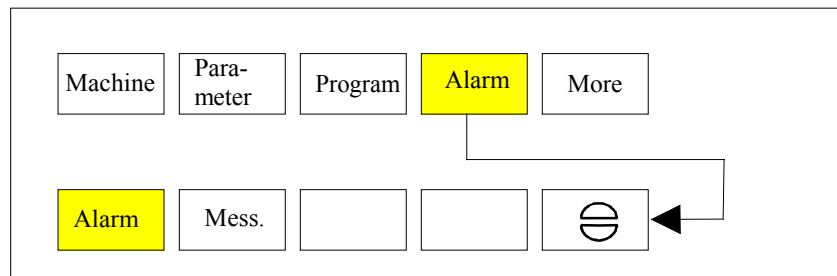
Parameter



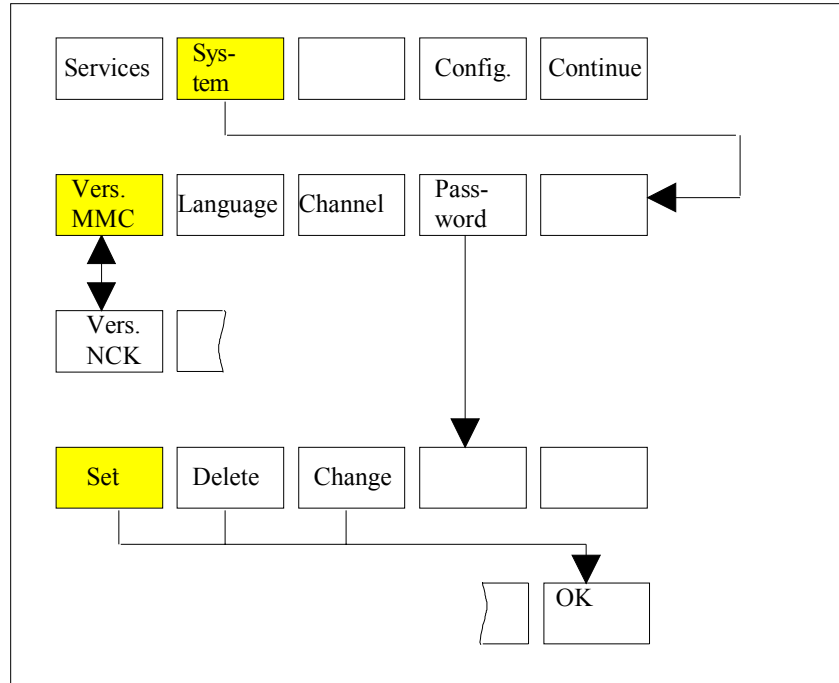
Program



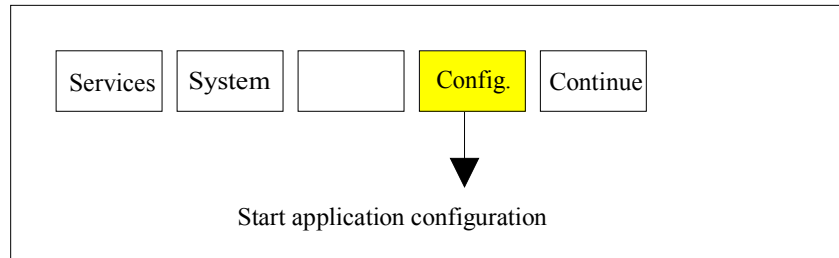
Alarm



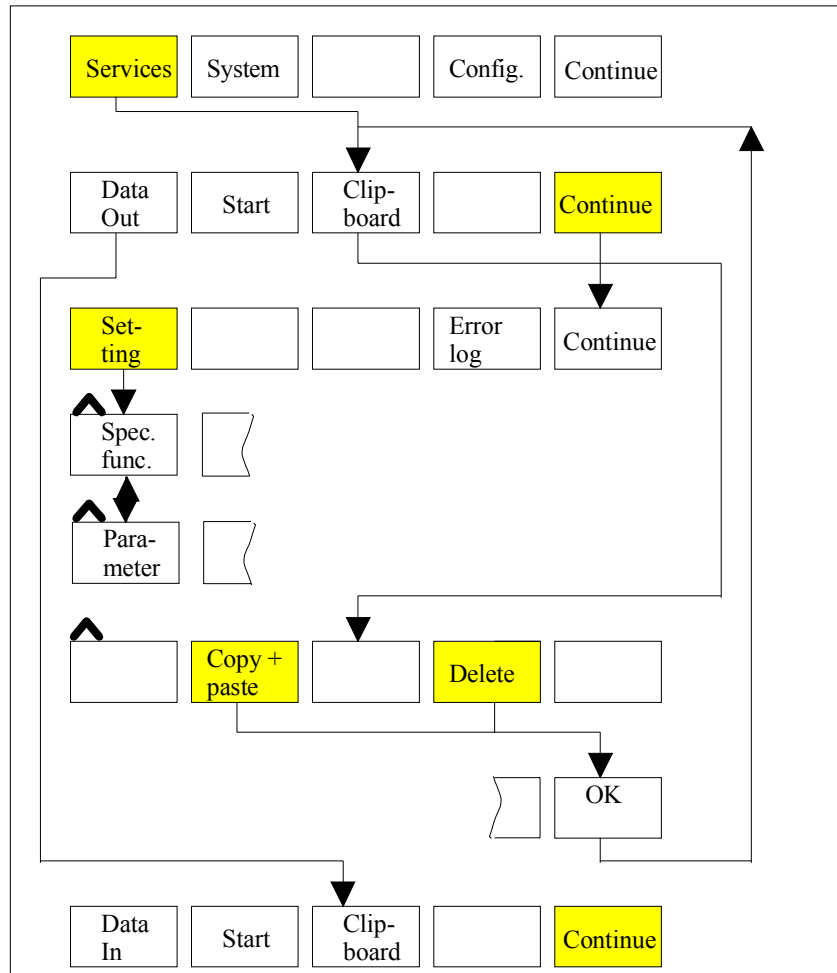
System



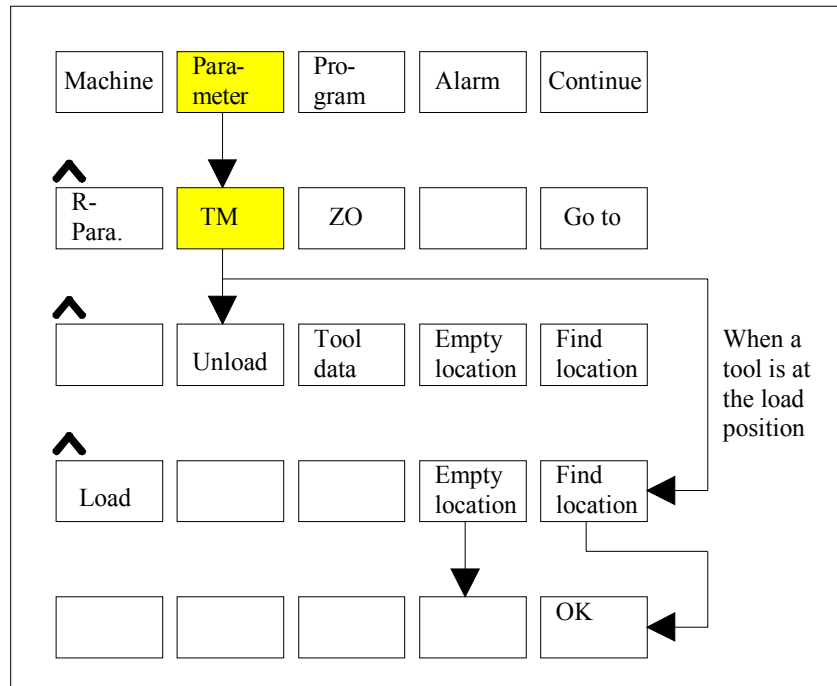
Configuration



Services



Tool management



Operation

3

3.1	User areas – Main display	BA/3-20
3.2	Actual value display	BA/3-21
3.3	Current block display	BA/3-22
3.4	Program control	BA/3-22
3.4.1	Program control – Single block	BA/3-23
3.5	Input and display of R parameters	BA/3-24
3.6	Work offsets: Overview.....	BA/3-24
3.7	Work offsets: Translation, rotation.....	BA/3-25
3.8	Work offsets: Scale and mirror	BA/3-25
3.9	Tool overview.....	BA/3-26
3.10	Tool offsets	BA/3-28
3.11	Tool management.....	BA/3-28
3.11.1	Load tool	BA/3-30
3.11.2	Unload tool.....	BA/3-31
3.11.3	Display and modify tool data	BA/3-32
3.11.4	Find empty location.....	BA/3-33
3.11.5	Select location in current magazine	BA/3-34
3.11.6	Select tool list to load	BA/3-34
3.11.7	Create new tool.....	BA/3-35
3.12	Workpiece overview.....	BA/3-36
3.13	Overview of the global main programs	BA/3-37
3.14	Overview of the global subroutines.....	BA/3-37
3.15	Display window	BA/3-38
3.16	Program selection.....	BA/3-38
3.17	Alarm and message overview.....	BA/3-39
3.18	System	BA/3-39
3.19	Read in data.....	BA/3-40

3.1 User areas – Main display

Display

- Press the User Area key to display the options for selecting user areas.
- The user areas appear on the softkeys.
- Press the "Continue" softkey to display further selection options.
- When you select a user area, the name of the user area appears in the top left-hand corner of the screen.
- Press the User Area key again to switch between the two most recently selected user areas.



Fig. 3-1 User area bar



Fig. 3-2 User area bar – Continue

3.2 Actual value display

- Display** of
- Axis name
 - Actual value
 - Unit
 - Residual path.

- Switchover** between
- Work and machine
 - Large and small actual value display

Machine	1	Jog	Reset	1120↓
WCS	Act val	D-to-go		
+A	0.000 mm	0.000		
+B	0.000 mm	0.000		
+C	0.000 mm	0.000		
+	0.000 mm	0.000		
+	0.000 mm	0.000		
MCS	Zoom	Curr. block	Prog-ctrl	

Fig. 3-3 Actual value display

- Actual value large** Switchover between machine and work via small actual value display

Machine	1	Jog	Reset	1120↓
MCS	Act val			
+ X	0.000 mm			
+ Y	0.000 mm			
+ Z	0.000 mm			
Act. vals	Zoom	Curr. block	Prog-ctrl	

Fig. 3-4 Large actual value display

3.3 Current block display

Display

of

- Program name
- Previous block
- Current block
- Next block in the execution sequence

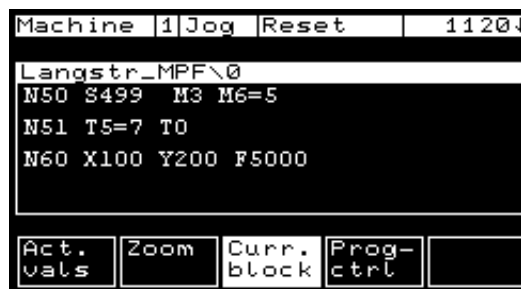


Fig. 3-5 Current block

3.4 Program control

Selection

- SKP Skip block, levels 0 to 7
- DRY Dry run feed
- ROV Rapid traverse override
- M01 Programmed stop
- DRF Select handwheel override
- PRT Program test mode



Fig. 3-6 Program control

Use the cursor and the Input key to select the boxes (left column).

A cross appears in the box to the right of the selected box as soon as the PLC has sent a message confirming that the function has been selected.

3.4.1 Program control – Single block

Selection

of

- SBL1 Single block with stop after every block that triggers a machine function.
- SBL2 Stop after every block.

Selection between display of

- all blocks or
- only traversing blocks.



Fig. 3-7 Program control – Single block

3.5 Input and display of R parameters

- Values displayed**
- R parameter index
 - Value of R parameters



Fig. 3-8 R parameters

3.6 Work offsets: Overview

Overview Overview of the existing settable zero offsets (scrolling possible with scroll bars).

Selection Display of a specific "settable zero offset" through positioning of the cursor and "Go to" softkey.



Fig. 3-9 Work offset overview

3.7 Work offsets: Translation, rotation

Display and input

- of the offset for geometry and special axes
- of the rotation angle for geometry axes
- of the selected G function for the work offset
- Accepting the zero offset data with Save, Recall or Area Switchover.
- Discarding the input data with Cancel.



Fig. 3-10 Work offset: Translation, rotation

3.8 Work offsets: Scale and mirror

Display and input

- of the scale factor
- of the mirror in a checkfield (select with Input)
- of the selected G function for the zero offset
- Accepting the zero offset data with Save, Recall or Area Switchover.
- Discarding the input data with Cancel.



Fig. 3-11 Work offset: Scale and mirror

3.9 Tool overview

Overview Overview of the existing tools (scrolling possible with scroll bars) with T number, tool identification number (tool name) and the tool offset (TO) area.

Selection Display of a specific tool offset through positioning of the cursor and "Go to" softkey.



Fig. 3-12 Tool overview

Find tool no. Enter number to be searched for and press the "Go to" softkey.



Fig. 3-13 Find tool no.

Tool management

Tools can be created and deleted by pressing the "Management" softkey.



Fig. 3-14 Tool management

New tool

Enter the new number and press the "OK" softkey to confirm.



Fig. 3-15 New tool

Delete tool

Enter the number to be deleted and press the "OK" softkey to confirm.



Fig. 3-16 Delete tool

3.10 Tool offsets

Display and input

- Display of the tool offsets for a tool cutting edge including the T number, D number and the tool identification number (tool name).
- All of the tool offset parameters can be displayed using the Page Up/Down keys.
- Switchover between the tool edges (up to nine edges) with the softkeys "D+" and "D-".
- Enter the new tool offset number.



Fig. 3-17 Tool offsets

3.11 Tool management

The OP 030 tool management comprises the following functions:

1. Load tool
2. Unload tool
3. Display and modify tool data
4. Look for empty location
5. Select location in current magazine and travel to loading location
6. Select tool list for loading
7. Create new tool

For further information about tool types, location types, duplo numbers and tool numbers, please refer to

Reference: /FBW/, Tool Management

Basic display

The following basic display is output when you select the "Tool Management" softkey:

Param.	1	Mda	Reset	1120↓
Duplo	MNr.	PNr.	PT	PZust
	1	5	0	-----F-
WZust			WGr	
Länge 1	Länge 2	Radius		
Be- laden			Leer- Platz	Suche Platz

Fig. 3-18 Status: A tool can be loaded

Param.	1	Jog	Reset	1120↓
Pippi1				
Duplo	MNr.	PNr.	PT	PZust
88	1	2	0	-----
WZust			WGr	1111
Länge 1	Länge 2	Radius		
0.000	0.000	0.000		
	Ent- laden	WZ- Daten	Leer- Platz	Suche Platz

Fig. 3-19 Status: A tool can be unloaded

The basic display for tool management functions specifies the data of the tool in the loading/unloading location. The display cannot be edited.

- First line: Tool identifier (max. 29 characters)
- Duplo: Duplo number (5 characters)
- MNo.: Magazine number (4 characters)
- LNo.: Location number in magazine (3 characters)
- LT: Location type (1 character)
- LStatus: Location status (8 characters):
- G – disabled
 - F – free
 - Z – reserved
 - B – reserved
- TStatus: Tool status (8 characters)
- A – active
 - F – enabled
 - G – disabled
 - M – measured
 - V – prewarning limit
 - W – changing
 - P – coded for fixed location
 - E – in use

- TGr.: Tool size (4 characters)
- Length 1 of tool (8 characters)
 - Length 2 of tool (8 characters)
 - Radius of tool (8 characters)

3.11.1 Load tool

Before a tool can be loaded, an empty location in the current magazine must be determined.

1. Select softkey "Find loc." or "Empty loc."
If no tool data are displayed except for the location number (LNo.), then the location concerned is empty.
2. Select the softkey "Load". The **Tool List** display appears on the screen.

Werkzeugbez.	Duplo	PNr.
Pippi0	4	99
Pippi1	5	88
Pippi2	0	0
Pippi3	7	66

Fig. 3-20 Load tool

The display shows the list of all the tools defined in the NC.

- Tool designation (max. 16 characters)
 - Duplo number (5 characters)
 - Location number (3 characters)
0 means that the tool is not contained in the magazine.
3. Use the cursor to select a tool and then select the softkey "OK".
The data are transferred to the tool management (i.e. the tool is now loaded).

A tool that is already present in the magazine (LNo. = 0) cannot be loaded again. If the desired tool is not defined in the list, you can create a new tool by selecting the softkey "New tool".

Notes

- The softkey "New tool" is activated only when an empty location is displayed.
- You can cancel the loading process by selecting the "RECALL" key.

3.11.2 Unload tool

1. Select the softkey "Unload".
An enquiry window requesting confirmation of the command to unload the tool from the current location is displayed.
The tool data are not deleted from the TO area.



Fig. 3-21 Unload tool

2. Select the softkey "Delete".
The tool is unloaded and the tool data are deleted from the TO area.
The basic display is output again when the tool has been unloaded.

You can cancel the unloading process by selecting the "RECALL" key.

Notes

- It is not possible to unload a tool from an empty location.
 - The softkey is activated only if there is a tool in the loading position.
-

3.11.3 Display and modify tool data

1. Select the softkey "Tool data".
You can modify the geometry data in the tool data display.

Param. 1 Jog Reset 1120↓				
Pippi1				
Duplo	88	WNr.	10	S 1
\	Geo	Verschl.	Basis	
L1:	0.000	0.000	0.000	
L2:	0.000	0.000	0.000	
R :	0.000	0.000	0.000	
<div style="display: flex; justify-content: space-between; border-top: 1px solid black;"> S neu WZ-Daten S Plus Ok </div>				

Fig. 3-22 Display/modify the tool data

The display shows additional data for the tool at the loading position.

- First line: Tool identifier (max. 29 characters)
- Duplo: Duplo number (5 characters)
- TNo.: Tool number (5 characters), internal tool number
- S: Cutting edge number (1 character). Can be incremented with softkey "C-edg plus".
- Geo: Geometry data defining length 1, 2 (L1, L2) or radius (R) (8 characters) can be edited (input fields).
- Wear: Wear data
- Base: Basic data

2. Make the appropriate data modifications and then select the softkey "OK".
The tool management basic display is then output.
You can abort the modifications by selecting the "RECALL" key.

Notes

- It is not possible to unload a tool from an empty location.
 - The softkey is activated only if there is a tool at the loading/unloading location.
-

3.11.4 Find empty location

1. Select the softkey "Empty loc.". The **Empty Location** display appears in which you can enter the magazine location type and the tool size.



Fig. 3-23 Find empty location

You can enter the following data in this display:

- Location type (1 character, input field)
 - Size in half-locations (left, right, top bottom: 1-character input fields (1..7))
2. Enter the correct values and then select the softkey "OK". The search for an empty location commences.
 3. If an appropriate location has been found, the loading position is approached. The display with the tool list appears on the screen. If a location cannot be found, the error message "Empty location not found" is output. You must then enter new values and start the search again.
 4. Select a tool with the cursor and select the softkey "OK". The data are transferred to the tool management. (The tool is now loaded.)
- You can cancel the empty location search by selecting the "RECALL" key.

3.11.5 Select location in current magazine

1. Select the softkey "Find loc.". The **Magazine List** display is then output. This list shows the empty and occupied locations in the magazine.

PNr.	Werkzeugbez.	Duplo
1	Pippi0	99
2	Pippi1	88
3		0
4	Pippi3	66

Fig. 3-24 Select location in current magazine

The display shows a list of all locations in the magazine. An empty location is identified by the duplo number 0.

2. Select a location with the cursor and select the softkey "OK".
3. The tool location is moved to the loading position. The basic display with the new tool location then appears on the screen.

You can cancel selection of the location by means of the "RECALL" key.

3.11.6 Select tool list to load

1. Select the softkey "Load". The **Tool List** display then appears.
2. Select a tool with the cursor and then select the softkey "OK". The data are transferred to the tool management. (The tool is now loaded.)

A tool that is already present in the magazine (LNo. = 0) cannot be loaded again.

If the desired tool is not defined in the list, you can create a new tool by selecting the softkey "New tool".

Notes

- The softkey "New tool" is activated only when an empty location is displayed.
 - You can cancel the loading process by selecting the "RECALL" key.
-

3.11.7 Create new tool

1. Select the softkey "New tool" in the tool list display.



Fig. 3-25 Create new tool

2. You need to enter the following data:
 - Tool identifier (16 characters)
 - Duplo number (1 character)
 - Tool type (3 characters)
 - Location type (1 character)
 - Size in half-locations: 1-character input fields (1..7) for left, right, top, bottom

The tool does not need to be assigned to a location yet. In some cases, it is only set up in the tool management memory and will be loaded to a location later on.

3. Your inputs are accepted when you select softkey "OK", the Tool Data screen appears.



Fig. 3-26 Tool data

The following additional data can be input:

- S: Cutting edge number
Can be incremented with the softkey "C-edge plus" (1 character). You can generate further cutting edge numbers (max. 9) with the softkey "New C-edge".
- Geo: Geometry data (8 characters)
- Wear: Wear data (8 characters)

Base: Base data defining
length 1 (8 characters),
length 2 or radius (8 characters)

4. Transfer your inputs to the system by selecting the softkey "OK". The tool list display appears. You have now set up the new tool. If you wish to load it, proceed as follows.
5. By selecting the softkey "OK" again, you load the tool you have just created.

You can cancel the loading process and return to the basic display by selecting the "RECALL" key.

3.12 Workpiece overview

Display

- of the list of existing workpiece files
- of a single workpiece file by selecting with the cursor and pressing the input key.

Selection

of the workpiece by positioning the cursor and pressing the "Select" key.



Fig. 3-27 Workpiece overview

3.13 Overview of the global main programs

Display

- of the list of global main programs
- of the main program by selecting with the cursor and pressing the Input key.

Selection

of a main program by positioning the cursor and pressing the "Select" key.



Fig. 3-28 Overview: Global main program

3.14 Overview of the global subroutines

Display

- of the list of global subroutines
- of a single subroutine by selecting with the cursor and pressing the Input key.

Selection

of a subroutine by positioning the cursor and pressing the "Select" softkey.



Fig. 3-29 Overview: Global subroutines

3.15 Display window

Display

- of the selected workpiece
- of the selected global main program
- of the selected global subprogram.



Fig. 3-30 Display window

3.16 Program selection

Selection

- of the current workpiece
- of the current global main program
- of the current global subroutine.

Program selection is possible only in the Reset state.



Fig. 3-31 Program selection

Selection of a workpiece/program becomes active if access rights, machine state, etc. do not prevent it. Otherwise, an error message is triggered.

In both cases, the currently selected program is displayed.

You can close the window by selecting the "OK" softkey.

3.17 Alarm and message overview

- Selection**
- Display of the current alarms and messages in full text format (scrollable).
 - Switchover between alarm overview and message overview.
 - Cancelling of alarms with the softkey displaying the Cancel symbol.



Fig. 3-32 Alarm and message overview

3.18 System

- Selection**
- Switchover between
 - display of the OP 030 software version ("Vers MMC") and
 - display of the versions of all components contained in the NC control (scrollable) ("Vers NCK")
 - Switchover between two languages ("Languages")
 - Switchover between various channels ("Channel")
 - Select "Password" function



Fig. 3-33 System

- Password**
- Set, delete or change via "Password" softkey.
 - The current authorization is displayed.



Fig. 3-34 Password

Only passwords consisting of numerical characters can be changed here since the OP 030 features a numeric keypad only.

If the Password function is to be used, a numeric password must be defined during NCK start-up.

3.19 Read in data

You can read in data from an external device via the RS-232 interface.

When the Services user area (extension of basic menu) has been selected, the following basic display appears:

Basic display



Fig. 3-35 Basic display: Read in data

Basic display, continued

Press the "Continue" softkey to display the extended menu:



Fig. 3-36 Basic display: Read in data, extended menu

Parameterize interface

The RS-232 interface and the external backup device must be adjusted to each other.

The default settings are read from the BD_OP030.TEA file, which can be edited during SW update.

Reference: /FBO/, IK, Screen Kit

Press the "Settings" softkey to open the following input forms:



Fig. 3-37 Parameterize interface



Fig. 3-38 Special functions

These screen forms contain the default values and values already defined by the user. For changing the values position the cursor onto an input field and press the selection key repeatedly until the required value is displayed.

Use the "+" and "-" keys to change the value in the "End of transmission" field.

Note

Caution! Switching off the power supply results in a loss of the parameter assignments entered.

Start/Stop

Press the "Start" softkey to display the "Target directory" window.

Three different procedures are possible.

1. Path/workpiece from archive file
The directory paths for the files read in are also stored.
2. Nothing selected
All files are copied to the directory previously selected with the cursor irrespective of the directory paths.
3. Copy to clipboard
All files are copied to the clipboard irrespective of the directory paths.

Specify target directory

Use the cursor and the Input key to make the selection required and press the "OK" softkey to confirm.



Fig. 3-39 Specify target directory

Then the "Transmission in progress" screen is displayed.



Fig. 3-40 Transmission in progress

Paste from clipboard

If the data have been copied to the clipboard, press the "Copy + paste" softkey to store the data in the directory previously selected with the cursor.



Fig. 3-41 Clipboard: Copy and paste

Files can be removed from the clipboard by pressing the "Delete" softkey.



Fig. 3-42 Remove data from clipboard

■

Index

4

A

Actual value display 3-21
Addition 2-9, 2-10
Alarm 2-12, 2-15
Alarm overview 3-39
Area switchover key 2-8
Area Switchover key 2-13

B

Backspace key 2-8, 2-10

C

Calculator functions 2-10
Channel message line 2-12
Channel status display 2-11
Configuration 2-16
Control 1-4
Current block 3-22
Current block display 3-22
Cursor 2-9

D

Deleting 2-10
Dialog line 2-12
Display window 3-38
Down Arrow 2-10

E

Edit key 2-9, 2-10
Edit mode 2-10
Empty location, find 3-33

G

Global main programs 3-37
Global subroutines 3-37

I

Input key 2-9, 2-10
Inserting 2-10

L

Location in current magazine, select 3-34

M

Machine 2-14
Main Display 3-20
Main program 3-37
Message overview 3-39
Minus 2-9, 2-10
Mirror 3-25

N

Navigation mode 2-10
New tool, create 3-35

O

Objective 1-5
Operating mode display 2-11
Operator panel 2-8
Overwriting 2-10

P

Parameter 2-14
PLC alarms/messages 1-4
Plus 2-9, 2-10
Preconditions 1-4
Program 2-15
Program control 3-22
Program selection 3-38

R

R parameters 3-24
Read in data 3-40
Recall 2-8, 2-13
Rotation 3-25

S

Scale 3-25
Screen brightness 1-5
Screen partitioning 2-11
Scroll bar 2-9
Scrolling 2-9
Selection key 2-9
Services 2-17, 2-18
Single block 3-23
Softkeys 2-13
Software update 1-4
Subroutines 3-37
Subtraction 2-9, 2-10
System 2-16, 3-39

T

Target group 1-5
Tool data, display/modify 3-32
Tool list, select to load 3-34
Tool management 3-28
Tool offsets 3-28
Tool overview 3-26
Tool, load 3-30
Tool, unload 3-31
Translation 3-25

U

Undo 2-10
Up Arrow 2-10
User area bar 3-20
User area bar - Continue 3-20
User area display 2-11

W

Work offset overview 3-24
Workpiece overview 3-36



SINUMERIK 840D/810D

Configuring the OP 030 Operator Interface

Development Kit (EU)

Introduction.....	EU/1-3
1.1 Structure of the documentation.....	EU/1-4
1.2 Use.....	EU/1-4
1.3 Procedure	EU/1-5
Installation.....	EU/2-7
2.1 Software update: Basic information	EU/2-8
2.2 HW and SW requirements.....	EU/2-8
2.3 Setup of the configuration environment for the OP 030.....	EU/2-9
2.4 What is stored where?.....	EU/2-11
2.5 Makefile and *.mak files.....	EU/2-15
First Steps in Configuring	EU/3-19
3.1 First complete compilation	EU/3-20
3.2 First changes to application configuration	EU/3-22
3.3 Text principle – Modification and extension.....	EU/3-24
Extending the Application Configuration	EU/4-29
4.1 Important files in your OP 030 configuration environment.....	EU/4-30
4.1.1 Binary configuration file – proj.dat.....	EU/4-30
4.1.2 Configuration source file	EU/4-30
4.1.3 Configuration include file – proj.h.....	EU/4-31
4.1.4 Application makefile – app.mak	EU/4-31
4.1.5 Configuration data makefile	EU/4-31
4.1.6 Text files	EU/4-32
4.1.7 Text directory – makefile.....	EU/4-32
4.1.8 System list directory file – sy_l_dir.h	EU/4-32
4.1.9 Application list directory file – ap_l_dir.h.....	EU/4-33
4.1.10 Standard list directory file – st_l_dir.h	EU/4-33
4.1.11 List identifiers for application area – ap_mwl.h.....	EU/4-34
4.1.12 List identifiers for standard user area – std_mwl.h	EU/4-34
4.1.13 Notepad entry defines – nb_app.h.....	EU/4-35
4.1.14 Layout – Size definitions for fonts, windows, softkeys - layout.h	EU/4-35
4.1.15 Keyboard events – key.h.....	EU/4-35
4.2 Templates for your application configuration	EU/4-36
4.3 Number range.....	EU/4-36
4.4 Color definitions	EU/4-37
4.5 Open window on softkey press in application area.....	EU/4-37

Installation and Delivery of the Application Configuration	EU/5-41
5.1 Installing the application configuration	EU/5-42
5.2 Creating / Delivery of the application configuration.....	EU/5-44
Working the Visual Workbench and the Development Kit	EU/6-45
6.1 Using Visual Workbench	EU/6-46
6.2 Development kit	EU/6-48
6.3 Known restrictions and incompatibilities	EU/6-49
6.4 Alternative to Visual Workbench.....	EU/6-49
OP 030 Operation	EU/7-51
7.1 OP 030 – Test mode on the PC.....	EU/7-52
7.2 Key assignment in PC mode.....	EU/7-53
7.2.1 Key actions	EU/7-54
7.3 PC simulation mode.....	EU/7-55
7.4 Emulation of OP 030 data.....	EU/7-56
7.5 PC MPI mode.....	EU/7-57
Index	EU/8-59

Introduction

1

1.1	Structure of the documentation.....	EU/1-4
1.2	Use.....	EU/1-4
1.3	Procedure	EU/1-5

1.1 Structure of the documentation

The description of the functions for configuring the user interface for the SINUMERIK OP 030 is subdivided into the following documents:

- **Configuring Syntax**
General description of the configuring syntax.
- **Development Kit (Configuring Syntax.)**
Description of the environment for the development of a customized user interface.
- **Operator's Guide**
Operator's Guide for the standard operating functions.
- **Screen Kit: Software update and configuration**
Guide for updating the software on a SINUMERIK OP 030.

1.2 Use

Target group	This documentation is intended for the machine tool manufacturer who wants to design his own user interface for the OP 030.
Objective	With the aid of the OP 030 Development Kit and the associated function description, the machine tool manufacturer is able to: <ul style="list-style-type: none">• Create a customized user interface for the OP 030.• Test this user interface on a PC.• Load and run the customized user interface on the OP 030.• Create a OEM specific master disk of the modified system for his own service assignments.
Dependencies	The version of the OP 030 Development Kit used must match the version of the SINUMERIK 840D/810D.

1.3 Procedure

The OP 030 is **configurable** to enable optimum customization of the user interface according to the particular requirements of the machine.

The OP 030 is supplied with **standard operating functions** (in binary format). The sources for these standard operating functions are contained in the **Configuring Kit (OP 030 Development Kit)**.

You can configure the system in one of two ways:

- **Extending the standard user area**
You can use the full range of functions of the standard configuration.
You can use all update versions of the standard configuration immediately.
The standard configuration contains a softkey for switching to a prepared initial window for application configuring. You can expand the system from this point.
- **Substitution of the standard user area**
You can customize the entire user interface. Templates for specific applications can be taken over from the standard configuration.

- Configuring language** A description of the configuring language structure can be found in:
Reference: /FBO/, PS, only online: Configuring Syntax (Configuring Package)
- Software updates** A description of how to update the software can be found in:
Reference: /FBO/, IK, Screen Kit
- NCK interface** The interface to the NCK is described in:
Reference: /LIS/, Lists
- Operation** A description of how to operate the OP 030 can be found in:
Reference: /FBO/, BA, Operator's Guide



2

Installation

2.1	Software update: Basic information	EU/2-8
2.2	HW and SW requirements.....	EU/2-8
2.3	Setup of the configuration environment for the OP 030.....	EU/2-9
2.4	What is stored where?	EU/2-11
2.5	Makefile and *.mak files.....	EU/2-15

2.1 Software update: Basic information

See documentation on Screen Kit.

2.2 HW and SW requirements

- Standard PC with a 80386 processor or higher(80486 recommended), 8MB main memory, 3 1/2 inch disk drive and a standard VGA graphics card (min. resolution 640x480 pixels)
- Memory requirements for the OP 030 environment (configuration, PC simulation, system for OP 030 hardware) approx. 6MB on hard disk.
- MS-DOS 5.0 or higher (or a similar DOS version)
- Windows 3.1 or Windows for Workgroups or W9x
- Microsoft Visual C/C++ V1.0 or V1.52 (can no longer be purchased from Microsoft but, if required, a rudimentary version without the Workbench can be obtained from Siemens).

- Main memory:
The OP 030 PC simulation requires a minimum of 500KB of conventional main memory (in the area between 0–640KB).

If you are running the OP 030 PC simulation with an MPI interface on an NCK, this free memory must be available **after** the communications driver has been loaded (by calling "com030.bat", see below).

2.3 Setup of the configuration environment for the OP 030

Procedure Setup process

- If necessary, correct your **LASTDRIVE** entry in the "config.sys" to drive "L" or higher. (LASTDRIVE=Z)
- Edit the file "**autoexec.bat**":
Append the call
"**call <Ziellaufwerk>:\op030\op030bin\op030set**" to the end of the file.
This sets the access paths, include paths and environment variables for the OP 030 environment and executes
subst L: <Ziellaufwerk>\OP 030.
If you are uncomfortable with the pause at the end of op030set.bat, you can comment out the "PAUSE" command in the last line of the file.
- Edit the file <Ziellaufwerk>\op030\op030bin\op030set.bat:
If you have not used the standard installation path c:\msvc when installing Microsoft Visual C/C++ (MSVC), please correct the entry for the **MSVC tool directory**
"set TOOLROOTDIR= c:\msvc" (default)
in accordance with your installation.
- Reboot your computer to execute the commands in config.sys and autoexec.bat.

Initialization of the MS-Visual C++ Workbench

- Start MS-Windows and the MS-Visual Workbench.
- Add an entry for your DOS directory to the "**Executable Files Path**" in the **<Optionen> -> <Directories>** menu. Only then will the makefiles be executed correctly.
- Add the entry
;I:\public; I:\proj\h; I:\proj\app\h; I:\proh\std\h
to "**Include Files Path**" in the **<Optionen> -> <Directories>** menu.
The Workbench searches these directories in order to determine dependencies.

**Continuing to use
the Workbench
working
environment**

The Visual Workbench saves information about your current working environment (open files, search and replace character strings, etc.) in <project>.wsp files which are stored alongside your <project>.mak. Copy these files - but do not copy *.vcw and *.mak - from your previous configuration environment to the newly installed environment.

Copy the files you have modified and newly created files. To do this, refer to the files and directories listed in Section 4.1.

**Typical errors
during setup**

- Scenario:** Neither the Windows File Manager nor the DOS "deltree" command can delete the directory <Ziellaufwerk>\OP 030.
- Cause:** You still have a "substitute" on this drive.
- Remedy:** Exit Windows and delete the "substitute" with **subst /d**.
- Scenario:** Commands or files are not found.
- Cause:** You have probably appended a space at the end of the following set command in op030set.bat "set TOOLROOTDIR=<MSVC_DIR>".
- Remedy:** Delete the space at the end of the set command. Move to the end of the line using the <END> key on the cursor control pad.

2.4 What is stored where?

A brief overview of the relevant directories and files.
All paths refer to the virtual drive "L:".

In the \ directory

instutil	Directory for OP 030 auxiliary programs
bin	Run-time directory for OP 030 test.
public	Directory for system configuration includes.
proj	Directory for configuration source files.
dev_kit.txt	Readme file for the changes in the config.sys, autoexec.bat, op030set.bat.
op030set.bat	Sets the environment variables and paths for the OP 030 configuration environment.

In the \BIN directory

com030.bat	Starts the MPI drivers.
comoff.bat	Terminates the MPI drivers.
op030.exe	PC simulation for OP 030.
bt0.ini	Initialization file for the OP 030 start-up.
proj.dat	Binary configuration file - the configured user interface for OP 030.
proj_dat.ori	Backup copy of the supplied standard configuration file proj.dat.
*.sp?	Language text files for the OP 030.
mmc0_txv.ini	Initialization file for text and languages.
sim.ovl	Program for PC simulation mode.
bt0_con.cfg	Variable defaults for PC simulation.
*.dir (files and directories)	For PC simulation (workpiece and parts program overview).
bd_op030.tea	Display Machine Data: File with \$MM_... parameters for OP 030 configuration.

In the \PROJ directory

app	Directory for the sources for application configuration
std	Directory for the sources for standard configuration
text	Directory for texts
dat	Directory for linking the proj.dat
h	Directory for the configuration includes
makefile	Makefile for generation of the complete configuration (text, std, app, dat).

In the \PROJ\DAT directory

makefile	Makefile for linking the proj.dat and copying to \bin.
----------	--

In the \PROJ\H directory

ap_mwl.h	For application ID definitions
layout.h	Definitions for the layout
proj.h	Contains all system configuration includes, required for configuration.
std_mwl.h	For standard ID definitions.
t_gl.h	Text include for global text ID.
t_pj.h	Text include for application text ID.
t_ps.h	Text include for standard text ID.
ap_l_dir.h	List directory for application configuration.
st_l_dir.h	List directory for standard configuration.
sy_l_dir.h	List directory for standard configuration.

In the \PROJ\TEXT directory

pj	Texts for configuration.
al	Texts for alarms.
makefile	For generation of all texts.

In the \PROJ\TEXT\PJ and \PROJ\TEXT\AL directories

d	German texts (default foreground language)
g	English texts (default background language)
e	Spanish texts
f	French texts
i	Italian texts
makefile	For generation of the configuration texts\alarm texts.

In the directories \PROJ\TEXT\PJD , G , ...

pj.txt	Text source file for application configuration
ps.txt	Text source file for the standard configuration
gl.txt	Text source file for texts which are needed in the standard and in the application configuration
ala.txt	Text source file for the alarms for the application configuration
alm.txt	Text source file for the OP 030 alarms
asy.txt	Internal OP 030 system messages
aln.txt	Text source file for the NC alarms
*.sp1,2	Generated binary text files for foreground and background language.
makefile	For generation of the language-specific binary configuration text files (*.sp1,2).

In the \PROJ\TEXT\VALD , G , ... directories

alp.txt	Text source file for PLC alarm texts
alz.txt	Text source file for cycle alarms
alc.txt	Text source file for compile cycle alarms
*.sp1,2	Generated binary alarm text files for foreground and background language
makefile	For generation of the language-specific binary alarm text files (*.sp1,2).

In the \PROJ\STD directory

h	Include files for standard configuration (for function see documentation in the relevant file).
src	Configuration source files for the standard configuration (for function see documentation in the relevant file).
obj_c800	Objects and library for the standard configuration.

In the \PROJ\STD\OBJ_C800 directory

std.mak	Makefile for generation of standard configuration.
---------	--

In the \PROJ\APP directory

h	Include files for the application configuration.
src	Configuration source files for the application configuration.
obj_c800	Objects and library for the application configuration.

In the \PROJ\APP\H directory

app.h	Include file with defines and external, where appropriate Functions available from src\app.c.
-------	--

In the \PROJ\APP\SRC directory

app.c	Configuration source file with a Window and example configuration elements
temp_w.c	Template for child window
temp_m.c	Template for new user area.

In the \PROJ\APP\OBJ_C800 directory

app.mak	Makefile for generation of objects and library for application configuration.
---------	---

2.5 Makefile and *.mak files

*.mak files	<p>In the Visual Workbench, units which belong together (libraries, executables, etc.) are combined in projects. The generation rules and dependencies for these projects are stored in *.mak files - project makefiles. These *.mak files are managed by the Visual Workbench.</p>	
Makefiles	<p>in the context of the Visual Workbench are "external makefiles". The syntax and semantics of the makefiles are unknown to the Visual Workbench. The makefiles are executed both in the Visual Workbench and under DOS with the nmake command.</p>	
Opening projects	<p>Select <Open> from the <Project> menu in the Visual Workbench. <Open> Enter the appropriate settings for the file types under "List of Filetypes" - *.mak for *.mak files or *.* for makefiles. Click the desired *.mak or makefile to open the project. When you open a makefile project, Visual Workbench explicitly asks you to confirm that you want to load an external makefile.</p>	
The 5 most important makefiles	<p>\proj\makefile \proj\text\makefile \proj\app\obj_c800\app.mak \proj\std\obj_c800\std.mak \proj\dat\makefile</p>	<p>creates everything: texts and binary configuration file. creates all texts and text includes. creates the application configuration. creates the standard configuration. creates the binary configuration file from the application and standard configuration.</p>

The 5 individual make files are as follows:

\proj\makefile

This is an "external makefile". When this makefile is called, a generation run is triggered for the **entire configuration**.

The sequence of directories for the generation is as follows:

```
text    \proj\text\makefile
std     \proj\std\obj_c800\std.mak
app     \proj\app\obj_c800\app.mak
dat     \proj\dat\makefile
```

The output window contains all messages of the nmake calls in the processed directories.

\proj\text\ makefile

This is an "external makefile". When this makefile is called, a generation run is triggered for **all texts** (pj: configuration, al: alarms). The binary text files for the foreground and background languages are copied automatically to **\bin**.

The include files created with the foreground language (= master language) are copied to **\proj\h**.

\proj\std\ obj_c800\std.mak

This is a project makefile for the Visual Workbench. The complete generation of the **standard configuration** is possible with std.mak.

For this purpose, the text include files (\h\proj\t_*.h) must already have been generated. (The text include files contain the declarations for the text identifiers used during configuration.)

\proj\app\ obj_c800\app.mak

This is a project makefile for the Visual Workbench. The complete generation of the **application configuration** is possible with app.mak.

For this purpose, the text include files (\h\proj\t_*.h) must already have been generated. (The text include files contain the declarations for the text identifiers used during configuration.)

\proj\dat\ makefile

This is an "external makefile". It is used to call the link run for the binary configuration file - **proj.dat** -. Here std.lib and app.lib are linked to form a binary configuration file.

If the generation of the binary configuration file is successful, the following message appears:

```
"LINK : warning L4021: no stack segment
LINK : warning L4038: program has no starting address
PROJ.EXE - 0 error(s), 2 warning(s)"
```

The two warnings appear because of special link settings and can be ignored. The proj.dat binary configuration file is subsequently copied automatically to \bin.

- Expanding projects** The 'Application configuration' (\proj\app\obj_c800\app.mak) and 'Standard configuration' (\proj\std\obj_c800\std.mak) projects can be modified in the Visual Workbench by selecting <Edit> from the <Project> menu.
Please see the following sections for more information.
- Dependencies** The dependencies with include files, source files and libraries are already entered correctly for the "external makefiles".

In the project makefiles (*.mak) - std.mak and app.mak - the dependencies can be generated automatically with the <Scan Dependencies> item in the <Project> menu in the Visual Workbench.
Please see the following sections for more information.
- Without Visual Workbench** The configuration and texts can also be generated without the Workbench. For this, it is essential to use the same settings for environmental variables as when using the workbench. By calling "**nmake makefile**" or "**nmake -f std.mak**" each makefile executes the tasks described above. Because of the dependencies, it is only necessary to change to the I:\proj catalog and then initiate complete generation using the "nmake" command. All changed source files are included in the generation process, including the text files.

In order to delete all generation results, you can initiate the "make clean" command I:\proj catalog. To delete partial generation results from the configuration, change to the relevant subcatalog and execute this command as well.

After this it is advisable to generate everything anew, as described above.



First Steps in Configuring

3

3.1	First complete compilation	EU/3-20
3.2	First changes to application configuration	EU/3-22
3.3	Text principle – Modification and extension	EU/3-24

3.1 First complete compilation

- To check the setup process, first start the OP 030 software in PC simulation mode:
Command sequence:

```
L:  
cd \bin  
op030
```

Switch to application configuration with the "Area Switchover key" (**F10** on the PC) and the "**Continue**" and "**Config**" softkeys.

Only the "Application text" is displayed here.

Exit the PC simulation with the key sequence
<CTRL><x> (simultaneously) <ENTER>

- Start Windows and the Visual C++ Workbench.
- Select <Project> -> <Open> to open the makefile **L:\proj\makefile**.
- Start the complete compilation with <Project> -> <Build> **PROJ**.
Alternatively, without starting the Workbench, initiate the command "nmake" in the \proj catalog.

- If the complete compilation is performed correctly, the following message appears in the output window:

```
"LINK: warning L4021: no stack segment  
LINK: warning L4038: program has no starting address  
copy proj.dat l:\bin\proj.dat  
1 file(s) copied  
MAKEFILE - 0 error(s), 2 warning(s)"
```

The 2 warnings always occur when proj.dat is linked and can be ignored.

- A proj.dat and all *.sp1,2 files with the current date must now be located in the L:\bin directory. Only sytx.sp1,2 are not changed.
- Test the system by reloading the OP 030 software in PC simulation mode.

Typical errors

*"NMAKE: fatal error U1081: '\bin\textkonv': program not found
Stop."*

You have opened the makefile <Ziellaufwerk>:\OP030\PROJ\makefile.
You should always open the makefiles and projects from L:.

*"NMAKE: fatal error Include-file xxxxx not found
Stop."*

You have not added the entry required to the "Include Files Path" in the
<Optionen> -> <Directories> menu.

*"NMAKE: fatal error U1081: 'attrib': program not found
Stop."*

You have not added the entry for your DOS directory to the "Executable
Files Path" in the <Optionen> -> <Directories> menu.

*"NMAKE: fatal error U1081: 'xcopy': program not found
Stop."*

You have not added the entry required for your DOS directory to the
"Executable Files Path" in the <Optionen> -> <Directories> menu.

3.2 First changes to application configuration

- Start Windows and the Visual C++ Workbench.
- Select <Project> -> <Open> to open the project **L:\proj\app\obj_c800\app.mak**.
- Open the file app.c with the icon "Open a file of the current Project" (top left, below the 'File' menu). *Alternatively, without starting the Workbench, open the file using a standard DOS editor.*
- Search for the character string "TEXT".
- Duplicate the located line and modify the text object in the following way:
"TEXT (101, X_T_APP, Y_T_APP+15, T_PJ_DUMMY, CS_SMALL, 0, WHITE)".
In doing this, you output the text "T_PJ_DUMMY" on the screen a second time (displaced downward by 15 pixels).
- Start the compilation for the application configuration with <Project> -> <Build> **APP.LIB** or by clicking the middle Build icon. *Alternatively, initiate the command "nmake" in the \proj\app catalog.*
- The compilation is successful if the following message appears in the output window:

"APP.LIB - 0 error(s), 0 warning(s)"
- Select <Project> -> <Open> to open the makefile **L:\proj\dat\makefile** to link the binary configuration file proj.dat.
- Start the linking process for the binary configuration file with <Project> -> <Build> PROJ_DAT. *Alternatively, initiate the command "nmake" in the \proj catalog.*

- The compilation is successful if the following message appears in the output window:

**"LINK: warning L4021: no stack segment
LINK: warning L4038: program has no starting address
copy proj.dat l:\bin\proj.dat
1 file(s) copied
MAKEFILE - 0 error(s), 2 warning(s)"**
- A proj.dat with the current date and time should now be located in the L:\bin directory.
- Test the system by reloading the OP 030 software in PC simulation mode.- Switch to application configuration with the "Area Switchover key" (**F10** on the PC) and the "**Continue**" and "**Config**" softkeys. The text "application text" should now appear twice in your PC simulation.
- Terminate execution of OP 030.

Typical errors

Error:

L:\bin\proj.dat does not have the current date.
Your modification has not taken effect.

Cause of the error:

The PC simulation may have been running when PROJ.DAT was being recreated in the Visual Workbench. In this case, the simulation blocks the file L:\bin\proj.dat, which cannot then be updated. The Workbench output when linking PROJ.DAT then contains the error message:

```
Illegal SHARE operation          - L:\BIN\PROJ.DAT  
    0 files copied.
```

Remedy:

Exit the PC simulation and create PROJ.DAT again from the project L:\proj\dat\makefile.

3.3 Text principle – Modification and extension

Languages – multi-lingual text

The OP 030 can be equipped with two on-line languages. These languages are called the **foreground language** and **background language**.

The foreground language files have the extension ***.sp1** and the background language files have the extension ***.sp2**.

The foreground and background language of the OP 030 system can be substituted with other texts or languages without recompiling the configuration or the base system. However, the number of texts and their order during compilation of the texts may not be changed. This restriction does not apply to alarm texts - all text files which begin with an "a" (alm, aln, ala, alp, alc, alz).

Master language

The various texts in the binary text files must be compatible. Therefore there is always one master language for the overall text handling. This master language defines the number and order of the texts. In the OP 030 Development Kit **English** (`\proj\text*lg`) is set as the master language. Text source file

Configuration text source files are stored at the path `\proj\text\pj\d,g,...` (depending on the language).

Alarm text source files are stored in `\proj\text\pj\d,g, ...` and `\proj\text\al\d,g, ...` - this is where machine-specific alarms are stored (PLC alarms, cycle alarms and compile cycle alarms).

Text source files always have the extension ***.txt**.

Binary text files

The binary text files are generated with the aid of the text makefiles (stored at the **makefile** paths) and a text converter. They are stored alongside the text source files. The binary text files always have the extension ***.sp1** for texts used in the foreground language and ***.sp2** for texts used in the background language.

When the OP 030 is loaded, the `\bin` directory is searched for the binary text files.

The default settings of the configuration system are such that the text makefiles automatically copy the binary text files for the **English foreground language** (`\proj\text*lg*.sp1`) and the binary text files for the **German background language** (`\proj\text*d*.sp2`) to the **\bin directory**. If you prefer a different configuration, you should copy the appropriate files to the `\bin` directory. However, the `*.sp1` files always indicate the foreground language.

Text include files	<p>The text makefiles generate text include files from the configuration text source files so that the texts can be included in the configuration. The text makefiles copy the text include files to the <code>\proj\h</code> directory.</p> <p>The text include files define the symbolic identifiers (also called text IDs) for the configuration.</p> <p>t_pj.h Text include for application text IDs. t_gl.h Text include for global text IDs. t_ps.h Text include for standard text IDs.</p>
Text dependencies	<p>When changes are made to a pj.txt or gl.txt, the application configuration must be recompiled. When changes are made to a ps.txt or gl.txt, the standard configuration must be recompiled. This is handled by the corresponding dependencies in the makefiles.</p>
Text access – basic principle	<p>The assignments between text symbols and texts are made in configuration text source files. From this, the text makefile (+ text converter) generates a binary text file for the run-time system and a text include file for the Development Kit. The text include files are automatically included in the configuration source files by the configuration include file <code>proj.h</code>.</p>
Syntax of the configuration text source files	<p>The following three configuration source text files are stored at the paths <code>\proj\text\pj*</code>.</p> <p>pj.txt Configuration source file for the application configuration gl.txt General configuration source file ps.txt Configuration source file for the standard configuration</p> <p>The syntax of the configuration source files is as follows: <i> symb_txt_id "text" [// comment]</i></p>
Parameters	<p><i> symb_txt_id</i> Symbolic text identifier, must begin with a letter, max. length 45 characters. <i> symb_txt_id</i> must be unique within the entire configuration (standard and application). This is fulfilled by the following convention within the supplied configuration version: Text identifiers defined in the file XY.TXT always begin with 'T_XY_' and are unique within the file XY.TXT (check by search in editor). It is recommended to conform to this convention when adding new texts to the application.</p>

text Text to be displayed is enclosed by the characters """". A line break within the text to be output is defined by %n.

comment Is possible after the comment identifier // up to the end of the line.
The comment normally indicates the maximum length of text which can be displayed on the screen.

Example of text extension

- Start Windows and the Visual C++ Workbench.
- Select <Project> -> <Open> to open the language makefile **L:\proj\text\pj\d\makefile**. This is also the makefile for the master and foreground language used for displaying all texts after the start of the OP 030.
- Open the file **L:\proj\text\pj\d\pj.txt**. *Alternatively, without starting the Workbench, open the file using a standard DOS editor*
- Append the following text to the end of the file:
"T_PJ_HELLO "Hallo Welt" // 20 characters"
- Start the text makefile with
<Project> -> <Build> **TXT_PJ_D**. *Alternatively, initiate the command "nmake" in the \proj\text\d catalog.*
- The compilation is successful if the following message appears in the output window:
...
MAKEFILE - 0 error(s), 0 warning(s)"
- Select <Project> -> <Open> to open the language makefile **L:\proj\text\pj\g\makefile**. This is also the makefile for the background language, i.e. the language to which a changeover can be made in the "System" user area of the OP 030.
- Open the file **L:\proj\text\pj\g\pj.txt**. *Alternatively, open the file using a standard DOS editor .*
- Append the following text to the end of the file:
"T_PJ_HELLO "Hello World" // 20 characters"

- Start the text makefile with
<Project> -> <Build> **TXT_PJ_G**. Alternatively, initiate the command "nmake" in the \proj\text\g catalog .
- The compilation is successful if the following message appears in the output window:
...
MAKEFILE - 0 error(s), 0 warning(s)"
- Select <Project> -> <Open> to open the project
L:\proj\app\obj_c800\app.mak. Alternatively, open the file using a standard DOS editor.
- Open the file app.c by clicking the icon "Open a file of the current Project" (top left).
- Search for the symbolic text identifier "T_PJ_DUMMY".
- Change the symbolic text identifier:
"TEXT (101, X_T_APP, Y_T_APP+15, **T_PJ_HELLO**, CS_SMALL, 0, WHITE)".
This outputs the text "T_PJ_HELLO" on the screen.
- Start the compilation for the application configuration with
<Project> -> <Build> **APP.LIB**. Alternatively, initiate the command "nmake" in the \proj\app catalog .
- The compilation is successful if the following message appears in the output window:
"APP.LIB - 0 error(s), 0 warning(s)"
- Select <Project> -> <Open> to open the makefile
L:\proj\dat\makefile to link the binary configuration file proj.dat.
- Start the linking process for the binary configuration file with
<Project> -> <Build> **PROJ_DAT**. Alternatively, initiate the command "nmake" in the \proj\ catalog .
- The link is successful if the message which appears in the output window has the following ending:
"...
Make Complete.
MAKEFILE - 0 error(s), 2 warning(s)"

- A proj.dat and a pjtx.sp1 and pjtx.sp2 with the current date and time should now be located in the L:\bin directory.

- Test the system by loading the OP 030 software in PC simulation mode:
 1. Change to the application window:
 - "Area switchover key" (**F10** on the PC)
 - "Continue" softkey
 - "Config" softkeyThe text "Hallo Welt" should now appear.

 2. Select a different language as follows:
 - "Area switchover key" (**F10** on the PC)
 - "Continue" softkey
 - "System" softkey
 - "Language" softkey

 3. Change back to the application window (see above).
The text "Hello World" should appear.

 4. Exit the PC simulation.

Typical errors

Error:

*"ERROR Line 7: Symbol T_PJ_HELLO not contained in the foreground language!
Conversion terminated.
1 Errors"*

Note:

You should enter the text in the master language first (German in standard versions) and generate these binary text files. Now you can insert the text in the other language text source files.

Error:

Only "???" appears as the text.

Note:

You have not generated the texts for the foreground language (German in our case); *.sp? files are missing from L:\bin.

Error:

"Error in command line /S1 or /S2"

Note:

The file op030txv.ini does not exist in the relevant \bin directory.



Extending the Application Configuration

4

4.1	Important files in your OP 030 configuration environment.....	EU/4-30
4.1.1	Binary configuration file – proj.dat.....	EU/4-30
4.1.2	Configuration source file	EU/4-30
4.1.3	Configuration include file – proj.h.....	EU/4-31
4.1.4	Application makefile – app.mak	EU/4-31
4.1.5	Configuration data makefile	EU/4-31
4.1.6	Text files	EU/4-32
4.1.7	Text directory – makefile.....	EU/4-32
4.1.8	System list directory file – sy_l_dir.h	EU/4-32
4.1.9	Application list directory file – ap_l_dir.h.....	EU/4-33
4.1.10	Standard list directory file – st_l_dir.h	EU/4-33
4.1.11	List identifiers for application area – ap_mwl.h.....	EU/4-34
4.1.12	List identifiers for standard user area – std_mwl.h	EU/4-34
4.1.13	Notepad entry defines – nb_app.h.....	EU/4-35
4.1.14	Layout – Size definitions for fonts, windows, softkeys - layout.h	EU/4-35
4.1.15	Keyboard events – key.h.....	EU/4-35
4.2	Templates for your application configuration	EU/4-36
4.3	Number range.....	EU/4-36
4.4	Color definitions	EU/4-37
4.5	Open window on softkey press in application area.....	EU/4-37

4.1 Important files in your OP 030 configuration environment

4.1.1 Binary configuration file – proj.dat

Description The **proj.dat** file represents the main product of your efforts. proj.dat contains the entire configuration of the user interface in binary format. However this file does not contain the texts but only text references. The texts are read from binary text files.

This configuration file is generated from the **configuration source file** in a compiler and linker run.

4.1.2 Configuration source file

Description Your application area source files (**\proj\app\src*.c**) and the standard user area source files (**\proj\std\src*.c**) together form the configuration source files.

The files must have the extension *.c. The syntax must conform to the syntax defined in the following documentation:

Reference: /FBO/, PS, Configuring Syntax

The files must be stored at the path described above.

The application source files and the standard source files are stored together in an application library and a standard library with the aid of the compiler and the library manager. These two libraries serve as an input for the linker, which generates the binary configuration file (proj.dat) from them.

Note The **configuration include file** must be included in each configuration source file.

Example A configuration source file with the first application window and the preparation for a further child window is located in L:\proj\app\src\app.c.

L:\proj\app\src\templ_w.c is a template file for a child window underneath the application window.

Please copy this template and modify the copy (see 4.5).

4.1.3 Configuration include file – proj.h

Description The configuration include file - proj.h - contains all the **C preprocessor macros and C list definitions** required for the configuration.

Syntax #include "proj.h"

Note The configuration include file must be the first include file to be included in each configuration source file.

4.1.4 Application makefile – app.mak

Description L:\proj\app\obj_c800\app.mak:
Project makefile for the application configuration.
Do not modify this file manually, it is managed by Visual Workbench (Edit Project, Scan Dependencies) !

4.1.5 Configuration data makefile

Description L:\proj\dat\makefile:
Project makefile for linking the application configuration and standard configuration.
This makefile generates proj.dat and copies it to your L:\bin directory.

4.1.6 Text files

Description The text files are stored in the directories L:\proj\text\pj\d,g,e,f,i.

A distinction is made between the following three text files:

pj.txt In L:\proj\text\pj\d, g, e, f, i:
Texts for application configuration

ps.txt In L:\proj\text\pj\d, g, e, f, i:
Texts which are required in the standard configuration.

gl.txt In L:\proj\text\pj\d, g, e, f, i:
Texts which are required in both the standard configuration and the application configuration.

4.1.7 Text directory – Makefile

Description In L:\proj\text\al\makefile and L:\proj\text\pj\makefile:
In these two makefiles you can specify with the letters D, G, F, E, I which language directories are searched during the generation of configuration and alarm texts.

4.1.8 System list directory file – sy_l_dir.h

Description The system list directory file is located in L:\proj\h\sy_l_dir.h.
The system list directory contains all **list pointers** for the lists of the base system (for switching between the standard user area and the applications user area) which are included in the **binary configuration file - proj.dat**.
The starting point for the interpretation of the binary configuration file is the first configuration list in the system list directory.

Note Changes to the system list directory should only be made by experienced users.

4.1.9 Application list directory file – ap_l_dir.h

Description The application list directory file is located in L:\proj\h\ap_l_dir.h.
The application list directory ap_l_dir.h contains external descriptions for all the **list pointers** for the lists of the application user area which are included in more than one configuration source file.

Syntax **EXTERN_list type (id)** external list reference

Parameters *id* List identifier

4.1.10 Standard list directory file – st_l_dir.h

Description The standard list directory is located in L:\proj\h\ap_l_dir.h.
The standard list directory st_l_dir.h contains external descriptions for all the **list pointers** for the lists of the standard user area which are used in more than one configuration source file.
A start and end identifier is not required for this purpose.

Syntax **EXTERN_list type (id)** external list reference

Parameters *id* List identifier

4.1.11 List identifiers for application area – ap_mwl.h

Description

All of the list identifiers for all the configuration lists used in the application user area are contained in L:\proj\h\ap_mwl.h.

The list identifiers are listed as an enum symbol in an enum list; new symbols are appended in a similar way as the existing entries. These symbolic enum identifiers are then used in the configuration source files for the definition and call(s) of the list.

The starting range of the enum list may not be changed, since the list identifiers will otherwise collide with the standard user area.

See also:

Reference: /FBO/, PS, Configuring Syntax

4.1.12 List identifiers for standard user area – std_mwl.h

Description

All of the list identifiers for all of the configuration lists used in the standard user area are contained in L:\proj\h\std_mwl.h.

The list identifiers are listed as an enum symbol in an enum list. These symbolic enum identifiers are then used in the configuration source files for the definition and call(s) of the list.

The starting range of the enum list may not be changed, since the list identifiers will otherwise collide with the application area.

See also:

Reference: /FBO/, PS, Configuring Syntax

4.1.13 Notepad entry defines – nb_app.h

Description The areas for the notepad entries (HMI variables) of the standard configuration are defined in L:\public\nb.h.
L:\proj\app\h\nb_app.h contains the range reserved for the notepad entries for application configuration.
Insert your defines for the notepads used in ordered form in this file.
The notepad defines may not be defined as enums.

See also:
Reference: /FBO/, PS, Configuring Syntax

4.1.14 Layout – Size definitions for fonts, windows, softkeys – layout.h

Description L:\proj\h\layout.h contains the default settings for the sizes of windows, softkeys and fonts.

See also:
Reference: /FBO/, PS, Configuring Syntax

4.1.15 Keyboard events – key.h

Description The keyboard events which can be configured in reaction routines are defined in L:\public\key.h.

Also see:
Reference: /FBO/, PS, Configuring Syntax

4.2 Templates for your application configuration

L:\proj\std\src Standard user area templates

In this directory you will find the configuration for the standard user areas. The following lists provide examples of some of the source files for various applications.

Display elements:

Text output:	in most files
Scroll bars:	in most files
Output fields - O_FIELD:	istw.c, ist_mks.c, mistw_g.c
Single/multiple selection boxes:	pr_benf.c, pr_benf2.c
Program overview:	m_ubers.c, w_uebers.c
User area selection:	sy_be_sk.c
Alarm display:	alarm.c
Tables:	r_rpa.c, wkz_idx.c, wkz.c, npv.c, npv_idx.c

Variable accesses on NCK:

Actual values:	istw.c
R parameters:	r_par.c
Zero offsets, frames:	npv.c, npv_idx.c
Tool offsets:	wkz.c, wkz_idx
Current block display:	akt_satz.c

4.3 Number range

List identifiers

The number ranges for list identifiers are defined in the files L:\proj\h\ap_mwl.h and L:\proj\h\std_mwl.h.

List element identifiers

The identifiers for list elements (such as input/output fields, actions, reactions) must only be unique to the relevant configuration source file.

Event codes

The event codes defined in the elements (WATCH_EVENT, VALUE_EVENT, BIT_EVENT) of the event list must be between **10.000** and **19.999**.

Notepad entries

Depending on their use and validity, notepads from different number ranges are used. The subdivision of the number ranges is defined in the L:\public\nb.h and L:\proj\app\h\nb_app.h files, which also contains the symbolic defines for the notepad numbers used.

Local notepad entries can be used for all ranges and can be overwritten with entirely different contents.

Definition in L:\public\nb.h and L:\proj\app\h\nb_app.h

Global notepad entries for the OP 030 system software and **global notepad entries** for the standard configuration:

Not for use in application configuration

Definition in L:\public\nb.h

Global notepad entries for application configuration:

Definition in L:\proj\app\h\nb_app.h

4.4 Color definitions

Description **BLACK** and **WHITE** are defined for the OP 030.

4.5 Open window on softkey press in application area

Objective Open a separate new window by pressing a key in the prepared application window.

Procedure 1. Preparation

First make sure that the software version is correct by performing a complete build:

Open L:\proj\makefile
Select <Project> <Build PROJ>

If the build is performed correctly (see section 3.1), you can continue with the following steps and localize any faults (e.g. typing errors) more quickly.

Verify that the software version functions correctly:

Start the OP 030 in the PC simulation.
Use the Area Switchover to change to the application window
Softkey 1 still has no function.

2. Define a text for the new window

Open the file L:\proj\text\d\pj.txt

Duplicate the line with the entry T_PJ_TEMPLATE_W

In the duplicate, replace the character string 'TEMPLATE_W' with 'APP_SUB1' by selecting 'TEMPLATE_W', entering the new text APP_SUB1 in <Edit> <Replace> starting the individual replace operation with <Find Next> and replacing the text with <Replace> and <Find Next>.

Save pj.txt

Convert the text by performing another complete build:

<Project> <Build PROJ>

A new app.c is created in proj\app, because when a text in the master language (German 'd') is changed, the corresponding include file and its dependent c files are generated again.

In all other languages except 'd' an error occurs during the text conversion, because a new symbol has been introduced to the master language d. You can ignore this error for now. You can remedy the error now or later by adding the text Id 'T_PJ_APP_SUB1' to the pj.txt files in all language directories as described above.

3. Create the standard objects for a window from a template

A new file is created for the window to be generated and prepared identifiers are renamed for the objects for the new window.

Open the project L:\proj\app\obj_c800\app.mak

Open the file L:\proj\app\src\templ_w.c

To prevent accidental modifications to this template file, select the option 'Read Only' in the Open File dialog box (on the right, under the Help button).

Copy templ_w.c to a new file, e.g. **app_sub1.c** by selecting
<File> <Save As> app_sub1.c (in the L:\proj\app\src directory)

Replace the character string 'TEMPLATE_W' globally with 'APP_SUB1',
<Edit> <Replace>

If you perform the replace operations with individual prompting, you can see which object and list IDs are affected.

Save app_sub1.c

You have now generated and assigned identifiers to the basic elements of the window. These IDs - with the exception of the Text IDs edited in step 2 - must now be declared to the system (see below).

4. Add the new file to the app.mak project

Select <Project> -> <Edit>

Select **app_sub1.c** in the File Name window and add it to the 'Files in Project' list by clicking <ADD>.

Close the 'Edit APP.MAK' window with <Close>.

The dependencies of the files belonging to the project are regenerated.

Note: The Workbench only searches directories which have been entered under <Options> <Directories> during initialization of the Workbench. Unfortunately, if an include file is not found in this path, **no warning** is given.

Save this addition by closing the project app.mak - with <Proj> <Close>. Regrettably, an explicit 'Project Save' without 'Close' is not available and the saving philosophy of the workbench is not very clear.

Reopen the project.

5. Define the new identifiers

List IDs

Open the file I:\proj\h\ap_mwl.h
Move to the end of the file
Copy the block XXX_TEMPLATE_W
In the duplicate, replace the character string 'TEMPLATE_W' with
'APP_SUB1'
Save ap_mwl.h

List branch entries

Open the file I:\proj\h\ap_i_dir.h
Copy the line 'EXTERN_WINDOW (W_TEMPLATE_W) ',
so that it appears above the comment.
In the duplicate, replace the character string 'TEMPLATE_W'
- CAUTION: not W_TEMPLATE !! - with 'APP_SUB1'
Save ap_i_dir.h

6. Perform the first compilation on the new file

Check the syntax of your changes by compiling app_sub1.c:
Make app_sub1.c the active window (click on front).
Select <Project> <Compile File app_sub1>

The compiler run should report 0 errors and 0 warnings.

7. Include the new window in the existing application

Open the file L:\proj\app\src\app.c

Search for the character string TEMPLATE_W

Delete the comment characters immediately before and after the two lines

```
RC_CLOSE_WINDOW (205, KEY_F1, W_APP, LOCAL, 1)  
RC_OPEN_WINDOW (210, KEY_F1, W_TEMPLATE_W, LOCAL)
```

Replace the character string TEMPLATE_W - CAUTION: not W_TEMPLATE !! - with APP_SUB1

Save app.c and select <Project> <Compile File app.c>

The compiler run should report 0 errors and 0 warnings.

8. Perform the complete compilation

Open L:\proj\makefile and select <Project> <Build> PROJ
Alternatively, enter the command "nmake" in the \proj catalog.

The files istw.c and sy_be_sk.c in proj\std are recompiled because they are dependent on the modified .h files.

9. Run a test in the PC simulation

Start the OP 030 in the PC simulation.

Use the Area Switchover to change to the application window and test the function of the softkey 1 and the Recall key.



Installation and Delivery of the Application Configuration

5

5.1	Installing the application configuration	EU/5-42
5.2	Creating / Delivery of the application configuration.....	EU/5-44

5.1 Installing the application configuration

Requirements

- An OP 030 environment is installed on your PC (OP030set.bat).
- PC and OP 030 hardware are connected by a V24 cable: (modem eliminator, order no. 6FX2002-1AA01-[country code], see **Reference:** /Z/, Catalog NC Z, Accessories).
- Your application configuration is located in L:\bin\proj.dat
- If you want to use other languages instead of German/English on the OP 030, the appropriate directories (e, f, i) under proj\text\al and proj\text\pj must be run through during text conversion (see 6.2).

Installations process

The **installation** process transfers your configuration settings, texts, your **configured user interface** and, if necessary, the OP 030 system software (update) from your PC to your OP 030.

You can select the following components for transfer:

- | | | |
|---|----------------------------|--|
| - | System Configuration Files | (Settings for device parameters and communication addresses) |
| - | Alarm Texts | (PLC alarms) |
| - | User Interface Languages | (Two languages of your choice for the user interface) |
| - | Application Program | (proj.dat, the configured user interface) |
| - | System Software | (L:\flashbin*.bin, the complete base system of the OP 030) |

The selection of a component means that the components specified before it are also transferred to your OP 030. (This is necessary for reasons of integrity.)

Connect the OP 030 to your PC on COM1 or COM2.

On the OP 030:

Switch on the power supply to the OP 030 or initiate a Power-ON-Reset. The OP 030 runs up and for 5 seconds displays the initial screen with the message that the self tests have been concluded and the menu for further operator inputs.

Change to the update mode by entering the digit **6** on the OP 030 during those 5 seconds, if required.

Select a baud rate. (57.6KB is recommended.)

On your PC:

Exit Windows, because the transfer times are much shorter under DOS.

Change to the substitute drive L:\

Start the update process with

"install".

Follow the installation instructions for selecting a package, language and PC COM Port.

Select the same baud rate on the PC as on the OP 030.

5.2 Creating / delivery of the application configuration

Objective You intend to supply your customers with your application (and the OP 030 software) enabling them to perform a software update of their OP 030.
The **makedisk** batch file included in the development kit generates update disks including the **setup** and **install** procedures the customers need for installation.
The update disks also comprise the **mkalarm** procedure for converting and integrating customer-specific alarm texts.

- Requirements, preparations**
- You have installed an OP 030 environment on your PC with **Setup**.
 - Make sure that the **setup_op.bat** file is in the I:\ root directory. If this is not the case, copy this from the installation disk (archive).
 - Your application configuration is located in L:\bin\proj.dat
 - Text changes have been made to all language directories (d, g, e, f, i) under L:\proj\text\al and L:\proj\text\pj.
 - You have entered additional instructions for the user in I:\readme.txt and info_oem.txt.
 - You require one blank formatted 1.44MB disk.

Procedure The software to be supplied is in the L:\vz_sys directories and copied from there to disks.

- The delivery contains the
- OP 030 base system
 - proj.dat configuration
 - Binary language files in 5 languages
 - Alarm text source files in 5 languages as well as tools for converting modified alarm texts
 - System configuration files
 - Setup / Install routines
 - mkalarm (for conversion of alarm texts at the end customers)

Change to the substitute drive L:\ in the I:\instutil catalog and start the creation process with

"makedisk".

Follow the instructions for selecting the delivery components.



Working the Visual Workbench and the Development Kit

6

6.1	Using Visual Workbench	EU/6-46
6.2	Development kit	EU/6-48
6.3	Known restrictions and incompatibilities	EU/6-49
6.4	Alternative to Visual Workbench.....	EU/6-49

6.1 Using Visual Workbench

General remarks

Only make a few changes at a time.

Because of the modelled dependencies in the makefiles only the files which are affected are regenerated. You hardly save any time by regenerating the files at longer intervals. However, the causes of errors are more difficult to localize.

Reduce the risk of accidental editing.

Select the write-protect option in the File – Open window of Workbench if you only need to view a file or use it as a template.
Iconize files which you do not currently require. These are loaded again as icons when you restart Workbench. Icons are always available but are write-protected.

Compiler and linker options

The following compiler options are set in the *.mak projects:

```
/nologo /G2 /W3 /ALu /Gt1 /Od /D "_DEBUG" /I "L:\public"  
/I "L:\proj\h" /I "L:\proj\app\h" /I "L:\proj\std\h"
```

\proj\dat\proj.mak also contain the following linker options:

```
/NOE /NOI /SEG:800 /ONERROR:NOEXE /MAP /BATCH  
/EXE /FAR /PACKC
```

If these have been changed unintentionally, they should be reset under <Options> <Project>.

Errors, Warnings: specification of line numbers

In commands which extend across multiple lines in configuration source files, the line number of the line containing the closing parenthesis or the end of the statement is always specified in error or warning messages.

For example, it should not be assumed in a parameter list that the last parameter was the incorrectly passed parameter.

Efficient work method

Advantages and disadvantages of a .mak project

In internal makefiles *.mak you can step through the messages in an error or warning list with the <F4> key. The Workbench automatically positions the cursor in the line of the file containing the error.

For a complete generation of your application, however, you have to change repeatedly to an external makefile, because the workbench does not provide adequate support for complex projects with defined directory structures.

Advantages and disadvantages of a makefile project

All of the components of the application subtree (such as texts) to be updated are generated automatically from a single point.

\proj\makefile updates the entire application and transfers it to the run-time directory \bin. No intermediate steps are omitted.

The searching of the directory structure always requires a certain amount of time.

A slightly longer output must be checked for relevant warnings, errors cancel the operation.

Reduce the amount of changing between makefiles

One work method for changing between \proj\makefile (if you modify texts) or proj\dat\makefile (if you leave texts unmodified) and \proj\app\obj_c800\app.mak combines the above advantages and minimizes the disadvantages.

The last four edited projects are offered at the bottom of the menu when the <Project> menu item is selected.

Extending projects**Creating a new .c- file from a template:**

The temp_w.c template files with preparations for an underlying window in application configuration are located in \proj\app\src. See Section 4.5.

Adding a .c file to the application

Open the project \proj\app\obj_c800\app.mak.

Select <Project> -> <Edit>

Select the file to be added in the selection window and add it to the project by clicking <ADD>.

The dependencies of the files belonging to the project are regenerated.

Note: The Workbench only searches directories which are entered under <Options> <Directories> during initialization of the Workbench. Unfortunately, there is **no warning** if an include file is not found within these paths.

Regenerating the dependencies

As soon as you add another include statement to a c file you have to regenerate the dependencies.

Open the project to which the c file belongs.

Start <Project> <Scan All Dependencies>.

The dependencies are saved in the .mak file when you close the project or exit the Workbench.

Save projects

Unfortunately, the Workbench does not offer an explicit 'Project Save' command.

The occasions on which data are saved and the amount of data saved are not fully documented.

In order to save new data, dependencies, open file windows, etc., you should save your work after certain important steps (e.g. Project Edit) by closing and then reopening the project.

Editor settings The <Options> <Color> menu item in the Visual Workbench editor allows you to highlight the various elements (source code, comments, etc.) in the text in color.

Troubleshooting **Error documentation**
 The Workbench uses an output window which is overwritten the next time "Build" is called. However, you can save the contents of this window under a name of your choice with <File> <Save As>.

6.2 Development kit

Customizing the text conversion You can define the language directories searched for text conversion. The German master language must be generated. The generation of all 5 languages is set when the system is supplied.

Edit the \proj\text\al\makefile and \proj\text\pj\makefile:
 Search for the definition of 'DIRS':

```
# Language -Directories ....
DIRS = d g e f i
# DIRS = d g
```

Move the comment character # one line up. Only directories named in DIRS are searched.

6.3 Known restrictions and incompatibilities

Text editors AEDIT and VI (for DOS)

Some ASCII editors - particularly older ones - add control characters to the text in order to represent special characters (such as accents, TABs, etc.) from the upper area of the ASCII table.

These characters cause errors during conversion of the text.

This has been encountered with AEDIT and VI (but not XVI). These editors should therefore **not be used** for the creation and management of text source files *.txt.

It is **not** advisable to use Windows editors because the OP 030 operates with an OEM character set and the editors under Windows use an ANSI character set. Not all character codes in these two character sets are the same and it is therefore certain to produce undesirable results.

If, in spite of this, Windows editors need to be used, a tool supplied at the same time is required to convert the text file before the text converter is used.

Call: a2o.exe <ascii.txt>. <oem.txt>

6.4 Alternative to Visual Workbench

As an alternative to Visual Workbench, you can edit and regenerate the sources under DOS (DOS box with Win9x) using an editor of your choice. However, the restriction applies that text files are not to be edited using a Windows editor because of the different character sets in Windows and DOS.

If, in spite of this, Windows editors need to be used, a tool supplied at the same time is required to convert the text file before the text converter is used.

Call: a2o.exe <ascii.txt>. <oem.txt>



OP 030 Operation

7

7.1	OP 030 – Test mode on the PC.....	EU/7-52
7.2	Key assignment in PC mode.....	EU/7-53
7.2.1	Key actions	EU/7-54
7.3	PC simulation mode.....	EU/7-55
7.4	Emulation of OP 030 data.....	EU/7-56
7.5	PC MPI mode.....	EU/7-57

7.1 OP 030 – Test mode on the PC

Requirements

Check the \bin directory for the existence of the following files:

OP030.exe
bt0.ini
mmc0_txv.ini
proj.dat
bd_tea.acc
bd_OP030.tea
alatx.sp1,2 alctx.sp1,2 almtx.sp1,2
alntx.sp1,2 alptx.sp1,2 alztx.sp1,2
asytx.sp1,2 gltx.sp1,2
pjtx.sp1,2 pstx.sp1,2 sytx.sp1,2

The following directories and files are also required in \bin for **PC simulation mode**:

Directories:

bd cst cus mpf
spf syf syf.dir wks

Files:

bd.dir cst.dir cus.dir mpf.dir
root.dir spf.dir wks.dir
sim.ovl bt0_con.cfg

The following files are also required for **PC MPI mode**:

- In \bin






com030.bat comoff.bat
mpidos.exe mpimon.exe
bt_l7tsr.com

If you are working under MS-Windows, please open a DOS Box.

Use the "mem" DOS command to check if you have enough main memory available. With the MPI drivers loaded you should have approximately 470KB free.

7.2 Key assignment in PC mode

The OP 030 keys are represented as follows on the standard PC MFII keys:

	Description	Key on MFII
	Area Switchover key Displays the main menu	F10
	Recall key Recalls the previous menu when the symbol is displayed	F9
	Edit key For editing input fields	Insert
	Input key Completes the input	Return / Enter
	Selection/toggle key Selection key for default values in fields identified by this key symbol	"5" on numeric keypad without Num-Lock
Arrow keys, Page Up/Down, Home, End	Navigation pad Navigation in windows and fields	Arrow keys, Page Up/Down, Home, End
Digits, "."	Numeric keypad Input of digits	Digits, "."
Backspace, (next to "0")	Backspace key Deletes previous character	Backspace
Softkey1- Softkey5	Softkeys Menu control	F1-F5

7.2.1 Key actions

Navigation mode You can normally switch between the individual fields with the arrow and the Page Up and Page Down keys.

Edit mode Position the cursor on the field you want to overwrite.
Overwrite field contents Enter the digits in the field.
Before you enter the digits the contents of the field are deleted automatically by the system and the field is empty.

Edit mode Position the cursor on the field you want to overwrite.
Change field contents Open the field with the edit key.
The contents of the field are retained.
You can now use the arrow keys to position the cursor in the field in order to edit the contents.

Saving the input You can save the input using the input key or by exiting the field with the Up Arrow or Down Arrow keys.

Undo If you press the edit key again in edit mode, the data entered in your field are not saved.
The field is closed and you are returned to navigation mode.
The value entered in the field before you switched to edit mode reappears in the field.

Delete single characters In edit mode, you can use the backspace key to delete the character to the immediate left of the cursor.

Calculator functions: Addition, subtraction Depending on their configuration, the fields are equipped with a calculator function enabling addition and subtraction operations to be performed on the field contents.
Open the field in edit mode for insertion with the edit key.
Enter + or -.
Enter your second operator (value).
Complete the entry with the input key.
The result of the calculation appears in the field.

7.3 PC simulation mode

Description	In PC simulation mode, part of the data on the NC or PLC is emulated by a simulation program. This mode is used exclusively to test the layout of the user interface and the response of the menu control. The contents of the data can differ greatly from the actual data in real use.
Requirements	<p>See also the previous section.</p> <p>In file bt0.ini in \bin, change the entry pc_test to pc_test = 1.</p> <p>Caution! Please do not modify any other entries in this file or change the order of the entries.</p>
Calling the OP 030	<p>Change to the L:\bin directory.</p> <p>The OP 030 software is started in PC simulation mode by entering OP 030.</p>
Exit	<p>You exit the OP 030 software by entering the key combination: <Ctrl>+<X>, then <Return>.</p> <p>(Press and hold <Ctrl>, press <x>, then release both keys and press <Return>)</p>
Typical errors in PC simulation mode	<p>bt0.ini missing; pc_test not set to 1.</p> <p>proj.dat missing.</p> <p>*.sp1,2 files missing.</p> <p>sim.ovl missing.</p> <p>bd, mpf, wks, spf + *.dir missing.</p> <p>bd_tea.acc, bd_OP030.tea missing.</p>

7.4 Emulation of OP 030 data

Description Specific default values can be assigned to the NC variables and OP 030 notepad entries for PC simulation mode in the file \bin\bt0_con.cfg.

Syntax `0 d_type 'sy_id area_unit col line B_type num_lines' "value" [# comment]`

Parameters

d_typ Data type

3 char
4 unsigned
7 long
8 float
0x0F double
0x13 string

sy_id Syntax identification

130 NUMERIK (define NUMERIK8)
130 SIMODRIVE (define DRIVE8)
16 PLC (define PLC10) is not supported
at the moment. !!!!
20 OP 030 (define MMC0)

area_unit NUMERIK (area and decimal unit (one byte)):

01 NCK
33 ... 63 Mode Grp1 ... Mode Grp31
65 ... 95 Channel1 ... Channel31
97 ... 127 Axis1 ... Axis31
129 ... 159 TOA1 .. TOA31
161 ... 191 FDD1 ... FDD31
193 ... 223 MSD1 ... MSD31

col Column (decimal)
(see **Reference:** /LIS/, Lists)

Line Line (decimal)
(see **Reference:** /LIS/, Lists)

B_type Block type (decimal)
(see **Reference:** /LIS/, Lists)

num_lines Number of lines (decimal)
(see **Reference:** /LIS/, Lists)

value Value dependent on data type

comment Any comment up to the end of the line

Example

```
#Separator format Syld Area Col Row BlockType NumCol Value
0 4 '130 65 1 1 16 1' "4"#P_C_Y_numGeoAxes
0 4 '130 65 2 1 16 1' "1"# " _numAuxAxes
0 4 '130 65 3 1 16 1' "4"# " _numMachAxes
0 4 '130 65 4 1 16 1' "4"# " _numSpindles
0 4 '130 33 3 1 127 1' "1"#B_S_opMode
```

7.5 PC MPI mode

Description

The OP 030 software communicates with an NC or PLC in PC MPI mode. The logical response corresponds to the real conditions of the OP 030 hardware.

Requirements

You must use a PG programming device or PC with an AT MPI card.

In file bt0.ini in \bin, change the entry
pc_test to pc_test = 0.

Notice

Please do not modify any other entries in this file or change the order of the entries.

For more information see the previous section.

Calling the OP 030

Change to the L:\bin directory.

Load the MPI drivers:
com030 tsr

The OP 030 software is started in PC MPI mode by entering
OP 030.

Exit

You exit the OP 030 software by entering the key combination:
Ctrl+X then Return.

The MPI drivers are deinstalled by the command:
comoff.

**Typical errors in
PC MPI mode**

bt0.ini missing

pc_test not set to 0.

MPI driver not loaded.

MPI card not connected correctly.

proj.dat missing.

*.sp1,2 files missing.

bd_tea.acc missing.

bd_OP030.tea missing.

You have insufficient main memory. You generally have more free
memory without MS-Windows.

You can increase the environment memory in the DOS box properties.



8

Index

*

*.mak 2-15

A

Addition 7-54

ap_l_dir.h 4-33

ap_mwl.h 4-34

app.mak 4-31

Application generation 2-14, 2-16, 3-22

Application library 4-30

Application list directory 4-33

Application text file 4-32

B

Binary configuration file 2-11, 2-12, 4-31

Binary text files 3-24

C

Calculator functions 7-54

Calling the OP 030 7-55, 7-57

Character size 4-35

Complete compilation 2-12, 2-16

Complete Compilation 3-20

Configuration source file 4-30

Configuring language 1-6

Configuring syntax 1-4

D

Delete 7-54

Development kit 1-4

Documentation 1-4

E

Edit mode 7-54

Emulation of OP 030 data 7-56

Event 4-35

Event codes 4-36

G

gl.txt 4-32

I

Insert 7-54

Installation 1-4, 2-9

K

Key Assignment 7-53

key.h 4-35

L

layout.h 4-35

Library 4-30

List element identifiers 4-36

List identifiers 4-34, 4-36

M

Makefile 2-15

Master language 3-24

MS-Visual C++ Workbench 2-9

N

Navigation mode 7-54

nb_app.h 4-35

NCK interface 1-6

NCK variables 1-6

Notepad entries 4-36

Number range 4-34, 4-35

O

Operation 1-6

Operator's Guide 1-4

Overwrite 7-54

P

PC MPI mode 7-52

PC MPI Mode 7-57

PC simulation mode 7-52

PC Simulation Mode 7-55

pj.txt 4-32

proj.dat 4-30, 4-31

proj.h 4-31

Project 2-15

ps.txt 4-32

R

Requirements 1-4

S

Softkey size 4-35

Software updates 1-6

st_l_dir.h 4-33

Standard library 4-30

Standard list directory 4-33

Standard user areas 4-36

std_mwl.h 4-34

Subtraction 7-54

sy_l_dir.h 4-32

System list directory 4-32

T

Text 4-32

Text access - basic principle 3-25

Text conversion 4-32

Text generation 2-12, 2-16

Text include 2-12

Text include files 3-25

Text makefile 4-32

Text source 2-13

Text source file 3-24

Text source files 3-25

Texts 3-24

U

Undo 7-54

W

Window size 4-35



SINUMERIK 840D/810D

Configuring the OP 030 Operator Interface

Screen Kit: Software Update and Configuration (IK)

Software Installation	IK/1-3
1.1 SETUP	IK/1-5
1.2 INSTALL	IK/1-6
1.3 Changing alarm texts	IK/1-8
1.4 Settings in configuration files	IK/1-9
1.5 Information	IK/1-11
Languages.....	IK/2-13

Software Installation

1

Description	This guide is intended to help you update part or all of the software on your OP 030.	
Scope of supply	System Software Development Kit	1 disk (SYSTEM DISK) or 3 disks (DEVELOPMENT KIT)
Hardware requirements	<ul style="list-style-type: none"> • Standard PC with at least 80386 processor and RS-232 interface • Hard disk space required for System software approx. 5.5MB Development Kit additional 20MB • Operating system MS-DOS 5.0 or higher or DOS box in W9x • RS-232-C Cable, Order No. 6FX2002-1AA01-[length code], see Reference: /Z/, Catalog NC Z, Accessories 	
OP 030 Software components	<p>The OP 030 software consists of the following components:</p> <ul style="list-style-type: none"> – System Configuration Files (settings for communication addresses of the partner) – Alarm Texts (PLC user alarms) – User Interface Languages (two languages you defined for the operator interface) – Application Program (proj.dat, the operator interface configured) – System software (complete basic system of the OP 030) 	
Note on software update program	The software update program is an individual part of the system software (on the OP 030 hardware) which exchanges the software components listed above.	

Important!

This software update program is able to replace **itself** with an update version. If a transmission fault is encountered during this self-replacement process (power OFF, RS-232-C interruption, etc.), the OP 030 hardware is likely to be in a **non-operational state**, which can be repaired only by the manufacturer. Consequently, the self-replacement process of the software update program should not be started unless the version supplied on disk is newer than the version currently installed on the hardware.

The "INSTALL" procedure prompts for the version currently installed to establish the necessity for a self-replacement.

Development Kit

The Development Kit is used to modify individual components of the OP 030 software (texts and application program).

A subsequent replacement of the software (via INSTALL) uses the modified components and the rest of the procedure is analog to software replacement from the set of system disks.

Installation process

The complete software installation is performed in two steps:

1. SETUP

The **Setup** procedure

- copies the required update programs onto the hard disk of your PC
- creates the update environment
- starts the installation process if required.

2. INSTALL

The **installation** procedure transfers configuration settings, texts, the configured operator interface and possibly OP 030 system software (from the OP 030 update directory on your PC/PG) to the OP 030 hardware.

1.1 SETUP

The **Setup** procedure for the system software or Development Kit is identical. Depending on how the software has been purchased, the procedure either requires a disk or operates via the Internet.

Procedure using a disk

- Insert DISK 1 in the disk drive of your PC.
- Start the setup by entering the command "**SETUP_OP**".
- The update environment requires **approximately 20 seconds** to be created.

Procedure for the Internet

- Create a catalog of your choice, e.g. OP_SETUP on the hard disk of your PC.
- Copy or extract (if purchased directly from Siemens) the files in this catalog.
- Change the source (FROM) and also, if required, the target (TO), in the **env_data.bat** file using a standard editor, e.g.:

<code>_FROM_HD=c:</code>	from the drive
<code>_FROM_DIR=OPSETUP</code>	from the catalog
<code>_TO_HD=c:</code>	to the drive
<code>_TO_DIR=op030</code>	to the catalog

In this case, the source files in the c:\OP_SETUP catalog are copied to c:\OP030.

- Start the setup by entering the command "**SETUP_OP**".
- Follow the SETUP dialog to select your update directory and the packages to be installed.
- The update environment requires **approximately 20 seconds** to be created.

**Possible exceptions/
problems during
SETUP:**

If a directory "<target drive>:\OP030" is found during setup, it is renamed "<target drive>:\OP030.OLD" (after confirmation) or the setup procedure is aborted.

If both OP030 and OP030.OLD already exist under "<target drive>", the directories are not renamed. Please delete or rename those directories.

Important!

The maximum number of drives accessed must be set to (>L) in the CONFIG.SYS file (e.g. LASTDRIVE=Z).

1.2 INSTALL

Description

Install copies of your configuration settings, texts, configured user interface and, where appropriate, your update software from the OP 030 update software (from the OP 030 update directory on your PC) to your OP 030 hardware. It will be invoked automatically by SETUP but it may also be executed at a later point in time.

You can select the following components for transmission:

- System Configuration Files (Settings for communication addresses of partner)
- Alarm Texts (PLC user alarms)
- User Interface Languages (Two languages of your choice for the user interface)
- Application Program (proj.dat, the configured user interface)
- System Software (Complete base system of the OP 030)

The selection of one component means that the components listed before it are also loaded on your OP 030. (This is necessary for reasons of data integrity.)

**Availability of
components**

The **system configuration files** are available with default entries and **INSTALL** permits editing them if required.

The **alarm texts** are available with default entries (under \OP030\proj\text\al\LANGUAGE DIRECTORY*.txt). *.txt files can be modified in standard editors; the modified files must be converted into the internal OP 030 format (by using 'mkalarm', see next section) before **INSTALL** is invoked.

The **user interface texts** are available in the internal OP 030 format for English, French, German, Italian and Spanish. **INSTALL** permits any two of these languages to be combined as foreground and background languages.

The OP 030 **application program** and the OP 030 **basic system software** are available as binary files.

**Using the OP 030
Development Kit**

The following applies if you intend to perform the software update from an installed Development Kit:

- The **alarm texts** are not converted by mkalarm but by the makefiles contained in the development kit.
- The **user interface texts** are available as *.txt files and are also converted by the makefiles.
- The OP 030 application program '**proj.dat**' which was generated under L:\bin is transferred to the OP 030 hardware.

Procedure

Connect the OP 030 to your PC with the RS-232-C cable mentioned above to COM1 or COM2.

On the OP 030:

- Switch on the power supply to the OP 030 or activate the Power-ON-Reset. The OP 030 runs up and for 5 seconds the menu for further operator input appears.
- During the 5 seconds, you can change to update mode by entering the digit **6** on the OP 030.
- Select a baud rate (57.6 kBaud recommended).

On your PC:

- Change to the directory <Ziellaufwerk>:\OP 030 (when the DEVELOPMENT KIT is installed, this is substituted by L:\).
- Start the update by entering (select 1 from setup)

"INSTALL".

**Entries obligatory
for INSTALL**

- Select software components to be transmitted (see above).
- For text files: select foreground and background language.
- For transmission of system software:
Specify whether the OP 030 hardware contains the current software version of the software update program.
- Select configuration file(s) to be edited if required.
- Select serial interface and baud rate for transmission.

1.3 Changing alarm texts

Alarm texts: Overview

The alarm text files *.txt are stored under
<Ziellaufwerk>\OP030\proj\text\all\<Sprachverzeichnis>
and the following name conventions apply to the language directories:

English	g	(master language, see below)	
German	d	French	f
Italian	i	Spanish	e

The files contain texts for the following alarm ranges:

alc.txt	Compile cycle alarms
alp.txt	Machine-specific PLC alarms
alz.txt	Cycle alarms

Modify alarm texts

All five language versions of each text file alc.txt, alp.txt, alz.txt must have the same structure (sequence of numbers, number of lines, etc.) as the "master language" (g).

The default texts contained in the *.txt files can be overwritten with user-specific text.

It is also possible to add new entries to the files. If new lines are added, they must always be added in the master language too before conversion (see below).

The standard DOS editor (EDIT) can be used to modify the *.txt files.

Inappropriate editors

Certain old ASCII editors may cause problems due to control sequences embedded in the text; such editors should not be used. The VI and AEDIT editors are known to cause such problems. Windows editors should **not** be used to edit the texts on account of the different Windows/DOS character sets (ANSI/OEM). If, in spite of this, Windows editors need to be used, a tool supplied at the same time is required to convert the text file before the text converter is used.

Call: a2o.exe <ascii.txt>. <oem.txt>

Convert alarm texts

For use on the OP 030 hardware, the alarm text files must be converted into alarm binary files.

Change to the directory <Ziellaufwerk>\OP030.

Enter **mkalarm** to start conversion.

The "mkalarm" procedure acts on all language directories and converts all *.txt files.

Any errors are listed in a log file, which is displayed on request.

Alarm numbers, syntax of alarm texts

For the subdivision of the alarm number ranges and the syntax of the alarm text files, please refer to the description given in the manual */IAD/*, Chapter 1, under '**Software and data**'

Reference: */IAD/*, SINUMERIK 840D Installation and Start-Up Guide

1.4 Settings in configuration files

Overview

The operation of the OP 030 is influenced by the following two files:

- **netnames.ini** Communications settings for the MPI bus partners
- **bd_op030.tea** 'Display machine data' (settings for operating modes via \$MM_parameters)

In general, the netnames.ini file does not require manual changes since the baud rate is set automatically by the OP 030.

The station address can be modified if required.

In the bd_op030.tea file it is possible to, for example, configure the RS-232 interface.

NETNAMES.INI

This file contains settings for the MPI/BTSS bus communication partners of your OP 030. In general, changes are not required in this field.

The line structure of the file is as follows:

```
<PARAMETER>= <WERT>
```

```
[own]
```

```
owner =          MMC_1
```

```
[conn MMC_1]
```

```
conn_1 =          NCU_1          Must not be changed!
```

```
[param network]
```

```
bus =             mpi          Must not be changed!
```

MPI/OPI bus station addresses:

Each station address must be assigned only once.

Addresses are permissible from 1 through 15.

```
[param MMC_1]
```

```
mmc_address = 10          Own station address on MPI/OPI bus
```

```
[param NCU_1]
```

```
nck_address = 13          NCK address on MPI/OPI bus
```

```
plc_address = 2           PLC address on MPI/OPI bus
```

If several OP 030s are to be operated at one NCU (NCK+PLC), it is necessary to change the own station address for the second OP 030 (e.g. from 10 to 11).

This file contains default settings for display machine data.

BD_OP030.TEA

The OP 030 evaluates the following parameters:

```
MM_LCD_CONTRAST
MM_DISPLAY_TYPE
MM_DISPLAY_MODE
MM_FIRST_LANGUAGE
MM_DISPLAY_RESOLUTION
MM_PRG_DEFAULT_DIR
MM_DISPLAY_BLACK_TIME
MM_TABULATOR_SIZE
MM_DARKTIME_TO_PLC
MM_SWITCH_TO_AREA
MM_PLC_HOTKEY

MM_USER_CLASS_READ_TOA
MM_USER_CLASS_WRITE_TOA_GEO
MM_USER_CLASS_WRITE_TOA_WEAR

MM_USER_CLASS_WRITE_TOA_ADAPT
MM_USER_CLASS_WRITE_ZOA
MM_USER_CLASS_READ_GUD_LUD
MM_USER_CLASS_WRITE_GUD_LUD
MM_USER_CLASS_OVERSTORE_HIGH
MM_USER_CLASS_WRITE_PRG_CONDIT
MM_USER_CLASS_WRITE_SEA
MM_USER_CLASS_READ_PROGRAM
MM_USER_CLASS_WRITE_PROGRAM
MM_USER_CLASS_SELECT_PROGRAM
MM_USER_CLASS_TEACH_IN
MM_USER_CLASS_PRESET
MM_USER_CLASS_CLEAR_RPA
MM_USER_CLASS_WRITE_RPA
MM_USER_CLASS_SEL_V24
MM_USER_CLASS_READ_IN
```

```
MM_USER_CLASS_READ_CST
MM_USER_CLASS_READ_CUS

MM_TRACE
```

The following parameters are used as a default setting for the RS-232 interface:

```
MM_V24_USER_XON
MM_V24_USER_XOFF
MM_V24_USER_EOF
MM_V24_USER_CONTROLS
MM_V24_USER_RTS Must not be altered!
MM_V24_USER_BAUD
MM_V24_USER_DATABITS
MM_V24_USER_PARITY
MM_V24_USER_STOPBIT
```

```
MM_V24_PRINTER_XON
MM_V24_PRINTER_XOFF
MM_V24_PRINTER_EOF
MM_V24_PRINTER_CONTROLS
MM_V24_PRINTER_RTS
MM_V24_PRINTER_BAUD
```

MM_V24_PRINTER_DATABITS
MM_V24_PRINTER_PARITY
MM_V24_PRINTER_STOPBIT

MM_V24_PG_PC_XON
MM_V24_PG_PC_XOFF
MM_V24_PG_PC_EOF
MM_V24_PG_PC_CONTROLS
MM_V24_PG_PC_RTS
MM_V24_PG_PC_BAUD
MM_V24_PG_PC_DATABITS
MM_V24_PG_PC_PARITY
MM_V24_PG_PC_STOPBIT

The following parameters are used as a default setting for tool management:

MM_TM_SINTDI
MM_TM_NUM_MAG
MM_TM_UNLOAD_AND_DELETE
MM_TM_LOAD_TOOL_NEW
MM_TM_TOOL_STATE_DEF_VAL
MM_TM_ACT_SEARCH_AND_POS
MM_TM-LOAD-LOC1
MM_TM-LOAD-LOC2
MM_TM-LOAD-LOC3
MM_TM-LOAD-LOC4
MM_TM-LOAD-LOC4
MM_TM-LOAD-LOC5
MM_TM_LOAD_PLACE

For application and range of values of these parameters, please refer to

Reference: //IAD/, Installation and Start-Up Guide, Section on Machine Data for Operator Panel

1.5 Information

Read the latest information for any software version in the attached file SIEMENS.D.TXT (German) or SIEMENSE.TXT (English).



Languages

2

Standard

In every edition the language files are provided in five languages and divided into

- **Alarm texts** <\OP030\proj\text\al\<Sprachverzeichnis>
- **Configuration texts** <\OP030\proj\text\pj\<Sprachverzeichnis>
- **User texts** <\OP030\proj\app\text\<Sprachverzeichnis>

Name convention for the language directories:

German	d	French	f
English	g	Spanish	e
Italian	i		

Special languages

The following languages are available as an option under Order Number **6FC5253-_AX00-_XG0 570887.9806.xx**

Dutch	n	Czech	z
Swedish	s	Hungarian	u
Portuguese	y		

As for system software, the installation is performed in two steps:

1. SETUP

The **Setup** procedure

- copies the required update programs onto the hard disk of your PC
- creates the update environment
- starts the installation process if required.

2. INSTALL

The installation procedure transfers texts from the OP 030 update directory (on your PC) to the OP 030 hardware.

Note

Windows editors should not be used to edit the texts on account of the different Windows/DOS character sets (ANSI/OEM). If, in spite of this, Windows editors need to be used, a tool supplied at the same time is required to convert the text file before the text converter is used.

Call: a2o.exe <ascii.txt> . <oem.txt>



SINUMERIK 840D/810D

Configuring the OP 030 Operator Interface

Introduction to Configuring (PSE)

Introduction	PSE/1-3
1.1 Structure of the documentation	PSE/1-4
1.2 Terms – Basic features	PSE/1-6
Hardware	PSE/2-13
2.1 Requirements	PSE/2-14
2.2 Overview	PSE/2-15
Configuration Tools	PSE/3-17
3.1 OP 030	PSE/3-18
3.2 MMC 100/HMI Embedded	PSE/3-19
Configuration Files	PSE/4-21
4.1 General	PSE/4-22
4.2 OP 030	PSE/4-25
4.3 MMC 100/HMI Embedded	PSE/4-26
Steps for Configuring the OP 030	PSE/5-27
5.1 User interface	PSE/5-28
5.2 Create application	PSE/5-28
Sequence for Configuring the MMC 100/HMI Embedded	PSE/6-31
6.1 User interface	PSE/6-32
6.2 Create an application	PSE/6-33
Compiler Error Messages	PSE/7-35
7.1 Errors	PSE/7-36
7.2 Warnings	PSE/7-37
Examples	PSE/8-39
8.1 Change an existing window	PSE/8-40
8.1.1 MMC 100/HMI	PSE/8-40
8.1.2 OP 030	PSE/8-45
8.2 Add window and open using softkey	PSE/8-46
8.2.1 MMC 100/HMI	PSE/8-46
8.2.2 OP 030	PSE/8-51
8.3 Input and output data	PSE/8-53
8.4 Results and reactions	PSE/8-58

Tips	PSE/9-65
9.1 General	PSE/9-66
Index	PSE/10-67

Introduction

1

1.1	Structure of the documentation	PSE/1-4
1.2	Terms – Basic features.....	PSE/1-6

1.1 Structure of the documentation

The description of the functions for configuring the user interface for the SINUMERIK OP 030 and MMC 100/HMI Embedded is subdivided into the following documents:

OP 030

- **SINUMERIK OP 030 – Configuring Syntax (PS)**
General Description of Configuring Syntax.
This description is delivered with the software and available as a pdf.
- **SINUMERIK OP 030 – Development Kit (EU)**
Description of the environment for the development of a customized user interface
- **SINUMERIK OP 030 – Operator's Guide (BA)**
Operator's Guide for the standard operating functions
- **SINUMERIK OP 030 – Screen Kit (IK)**
Guide for updating the software on a SINUMERIK OP 030

HMI Embedded

- **SINUMERIK 840D/810D – HMI Embedded Configuring Package**
Installation, configuring and operating

Target group	This documentation is intended for the machine tool manufacturer who wants to design his own user interface for the OP 030/MMC 100/ HMI Embedded.
Objective	Using the Development Kit and the associated function description, the machine tool manufacturer is able to: <ul style="list-style-type: none">• Create their own operator interface.• Test this operator interface on a PC.• Transfer the operator interface created to the target hardware and run it.• Create a vendor-specific master disk of the modified system for his own service assignments.
Dependencies	The Development Kit used must match the version of SINUMERIK 840D/810D.
Requirements	The hardware and software requirements for the use of the Development Kit are specified in the following document: Reference: /FBO/, EU, Development Kit /PJE/, HMI Embedded Configuring Package
Notation of the configuration language	The following grammar rules apply: <ul style="list-style-type: none">- " " separates alternatives (if not explicitly expressed as a logical combination),- "[" and "]" bracket optional components,- "..." identifies the optional repetition of parameters.- Parts printed <i>in italics</i> are user parameters.- Parts printed in bold characters are keywords.

1.2 Terms – Basic features

User interface	<p>The purpose of the user interface is to enable the user to monitor and control a machine tool.</p> <p>The machine tool is controlled by the NC (NCK and PLC). Indirect control of the machine tool is therefore possible by controlling the NCK and the PLC.</p>
NCK variables PLC variables MMC variables (notebook entry)	<p>Neither the NCK nor the PLC have a user interface which the operator can access directly.</p> <p>However, the NCK and the PLC do have a defined software interface.</p> <p>The system uses MMC variables which offer the machine tool manufacturer access to the internal variables of the MMC or to his own self-defined MMC variables. MMC variables are managed in notebooks.</p> <p>The objective of the machine tool manufacturer (using the Development Kit and Screen Kit) is therefore, using the NCK/PLC interface, to design a user interface which provides the closest contact with the machine tool.</p> <p>The PLC interface is generally defined by the machine tool manufacturer.</p> <p>The NCK interface is described in the following document:</p> <p>Reference: /LIS/, SINUMERIK 840D, Description of Functions, Lists</p>
Display elements and objects	<p>The layout of the user interface and the output of data on the screen are controlled with the aid of predefined display elements. The display elements are either static, i.e. unchanging (such as graphics composed of lines and rectangles) or dynamic (e.g. display of variables from the NCK).</p> <p>Display elements are also referred to on the following pages as display objects, window objects or simply objects.</p>
Dialog fields dynamic display elements	<p>Dynamic display elements dynamically change their state.</p> <p>The change can either be initiated by user inputs or by value changes in the NCK/PLC or MMC/HMI.</p> <p>Dialog fields are, for example, output fields, input fields, input/output fields, single/multiple selection fields, scroll bars, inverted fields and action fields.</p>

Process dialog, operation sequence	<p>The process dialog is controlled mainly by the user through direct interaction with the system.</p> <p>The process dialog can, however, also be influenced by machine tool states (implicitly via PLC and NCK).</p> <p>The user's main means of control are softkeys and the direct input of values.</p> <p>Reference: /BA/, SINUMERIK 840D/810D MMC Operator's Guide</p>
Softkey	<p>With the aid of the softkeys it is possible to organize the user interface in a hierarchical structure.</p> <p>A softkey is presented with a graphical part and a functional part – the latter are the softkey reactions.</p>
Window	<p>When structuring the user interface, static and dynamic display elements can be combined in a group. These groups are presented in windows.</p> <p>You can display (open) a window and the display elements grouped inside it or remove it from the screen (close it) by performing an action.</p> <p>The windows are usually opened or closed through user interaction by pressing a softkey.</p>
Menu	<p>Menus are also provided for controlling user interaction. A menu represents the starting point for a whole chain of windows with softkeys and softkey reactions. This produces implicit combinations of different windows in groups.</p> <p>Menus are closed when a new menu is opened.</p> <p>Global menu</p> <p>The global menu is called a header and cannot be changed. It is used to display data (e.g. operating mode, channel number) and always relates to all user areas.</p> <p>Local menu</p> <p>The local menu is the configurable part of the display and can be changed. The person performing the configuration work can create the interface and branch structure in the local menu. Only one local menu can be active at a time.</p>

User area

The first softkey level is called the main menu or data area bar. It generally reflects the main logical subdivision of the user interface. The individual local menus, which are hidden at this level behind the softkeys, are therefore also known as the user area. One menu is normally allocated to one user area.

The main subdivision into user areas can be displayed on the softkeys with the User Area Key at any time, regardless of the level of nesting where the user is currently located.

The allocation of the user areas to the softkeys on the OP 030 can be found in the file `\proj\std\sy_be_sk.c` (not MMC 100/HMI Embedded).

Actions and reactions

Values of NCK/MMC/PLC variables can be read, written, initialized, processed and calculated in action and reaction routines.

Reaction and action routines are assigned to a window.

Action routines are executed in transition phases, e.g. when menus or windows are opened or closed.

Action routines are used for initializing, saving and restoring variables and internal statuses.

Reaction routines are executed as a reaction to a user action (such as the press of a softkey or other key) or to changes in variable values (indirectly through events).

Events

The operating sequence is controlled by two factors: user actions (inputs) and status changes on the machine tool.

User actions are manifested in keyboard events (including softkey events and key presses).

The state of the machine tool can be supervised by means of monitoring functions (event list with BIT_EVENT, VALUE_EVENT, WATCH_EVENT). When the specified state occurs, a defined internal event is generated.

Actions can be triggered in response to the occurrence of both keyboard events and internal events – these actions are called reactions.

**Lists
configuration lists**

The configuring system is based on two management objects: the menu and the window.

Windows are assigned to one menu.

Actions, reactions, events, softkeys and display elements (objects) can be allocated to one window.

The number of these individual actions, reactions, events, softkeys and display elements is variable from menu to menu and from window to window.

A configuration list (also referred to as a list) is a collection of elements of the same type (e.g. actions, reactions, display elements).

A list is identified by:

- A list type (action list, reaction list, object list, softkey object list, limit list, etc.).
- A unique list identifier assigned by the person configuring the system.
- Beginning and end.
- A variable number of elements with different functionality (list elements).

Each list in the system can be identified uniquely by the list type and list identifier.

Important!

The list type must be a constant figure. Calculating operations are not permitted at this point.

List identifiers

Each list has its own list identifier.

This list identifier must be unique to the whole system and is assigned by the person configuring the system.

For more information on the list identifier, please refer to:

Reference: /FBO/, EU, Development Kit
 /PJE/, HMI Embedded Configuring Package

**List elements
list entries**

Each list consists of a variable number of the same or different elements, referred to as list elements.

These list elements are identified by their type and list-element-specific parameters.

Details of which elements (actions, reactions, limit values display elements) can be included in which lists are given with the individual elements.

Configuring data	<p>Configuring data is the term used to describe the compilable, linkable and thus interpretable user interface description created with the help of the OP 030 Development Kit, MMC/HMI Screen Kit, this documentation and the OP 030 configuring macros.</p> <p>The readable form of the data is stored in *.C files. The form which is processable by the OP 030 system is stored in the proj.dat (sl.dat) file in OP 030.</p> <p>Reference: /PJE/, HMI Embedded Configuring Package</p>
Standard user areas	<p>The user interface and the configuration data are organized in two groups: the standard user areas and the application user areas.</p> <p>For the OP 030 the configuration lists for these two areas are stored in different directories.</p> <p>OP 030 only: SIEMENS supplies the standard user areas both on the OP 030 and, in source form, in the Development Kit. These standard user areas can be customized by the machine tool manufacturer. It should be remembered when customizing the standard user areas that these areas are supplied again on the next system update.</p> <p>Reference: /FBO/, BA, SINUMERIK OP 030 Operator's Guide</p> <p>MMC 100/HMI Embedded only: The standard user areas are not supplied in source form and therefore cannot be customized by the machine manufacturer.</p>
Application user area	<p>The machine tool manufacturer has the option of extending or replacing part or all of the standard user areas with his own application user area.</p>
Compiling	<p>Configuration files (*.C files) created by the machine tool manufacturer are converted into binary format (*.OBJ) or library format (*.LIB) with the aid of a compiler.</p> <p>Reference: /FBO/, EU, Development Kit /PJE/, HMI Embedded Configuring Package</p>
Linking	<p>The linking process combines the binary configuration data of the standard user areas and application user areas.</p> <p>Reference: /FBO/, EU, Development Kit /PJE/, HMI Embedded Configuring Package</p>
Testing the configured user interface	<p>The user interface generated with the Development Kit can be tested on a PC in simulation mode.</p> <p>The final test, however, must be performed on the hardware.</p>

Texts foreign languages

The OP 030 or MMC 100/PCU20 and operator panel fronts have a feature which allows the user to switch between two languages (or between two other texts) on-line.

The texts are therefore stored separately from the configuration data.

Each text version (language) is stored in a separate directory.

For fast access, the texts are stored in binary format with the aid of a text converter.

Up to two text versions can be transferred to the target run-time system.

Reference: /FBO/, EU, Development Kit
/PJE/, HMI Embedded Configuring Package

Standard values for colors, window sizes, font sizes

The default definitions for the available colors, the character set used and the standard window sizes, etc. are described in the following documentation:

Reference: /FBO/, EU, Development Kit
/PJE/, HMI Embedded Configuring Package

Number ranges

The definition of the list areas, event areas and notepad areas used are described in the following documentation:

Reference: /FBO/, EU, Development Kit
/PJE/, HMI Embedded Configuring Package



Hardware

2

2.1	Requirements	PSE/2-14
2.2	Overview	PSE/2-15

2.1 Requirements

The hardware requirements are as follows:

- PC with 80386 processor (80486 preferred)
- 3.5" disk drive
- Standard VGA graphics card (min. 640 x 480 pixels)
- Min. 8MB main memory
- RS-232 interface cable
- MPI card with cable (for online simulation)

2.2 Overview

The following system constellations are possible for the SINUMERIK 840D/810D controls:

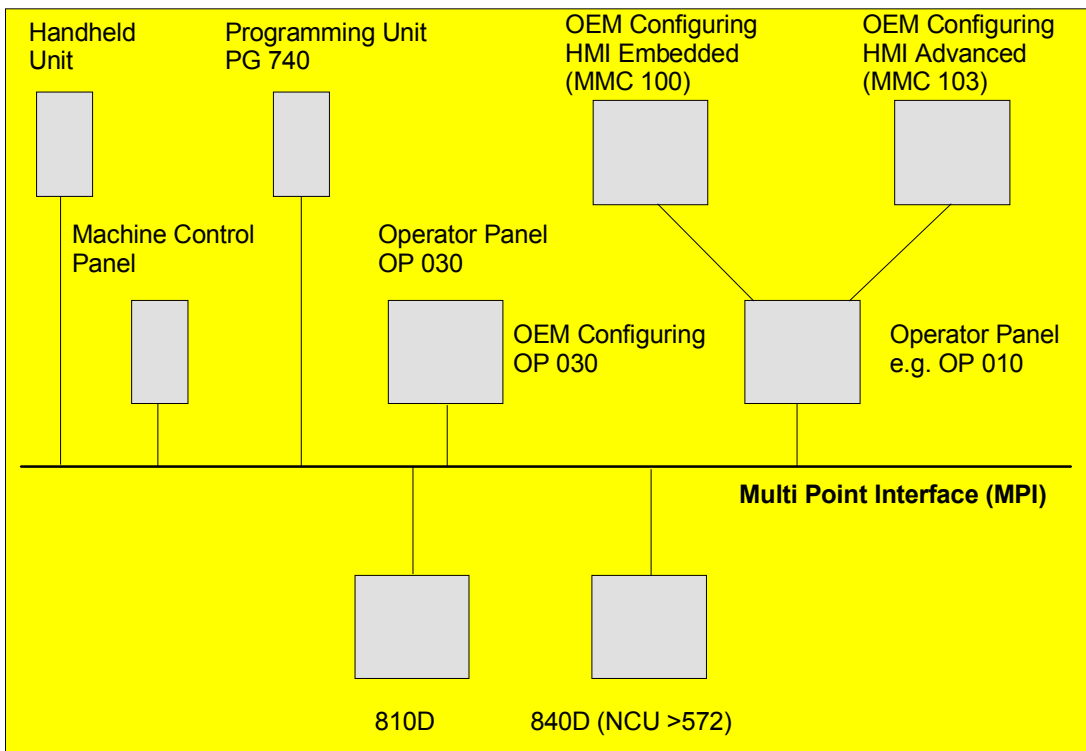


Fig. 2-1 System overview of SINUMERIK 840D/810D

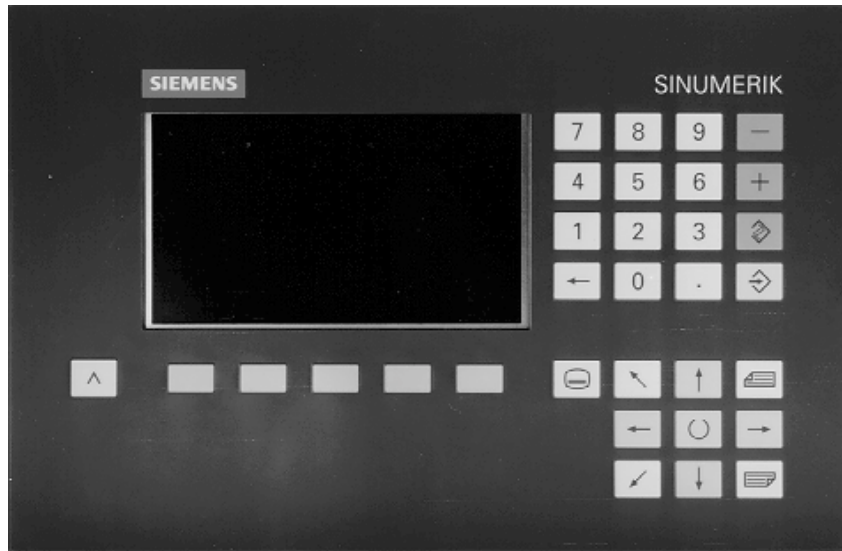


Fig. 2-2 OP 030 operator panel

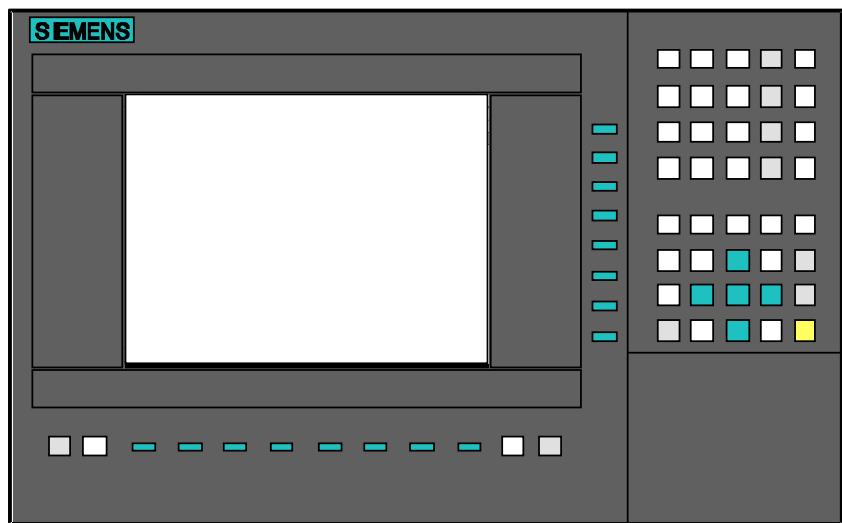


Fig. 2-3 MMC 100/OP 010 operator panel

■

Configuration Tools

3

3.1	OP 030	PSE/3-18
3.2	MMC 100/HMI Embedded	PSE/3-19

3.1 OP 030

The following tools are provided for configuring the OP 030:

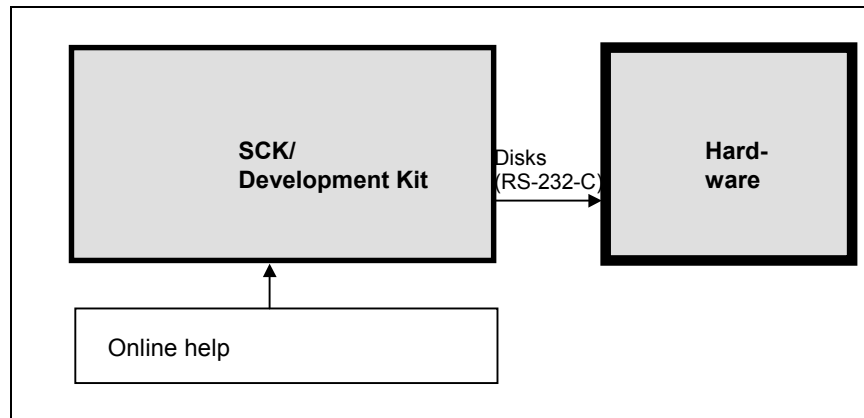


Fig. 3-1 Tools for OP 030

1. Development Kit

With this you can

- Create an application and/or change a standard application
- Create multilingual texts
- Change the machine configuration
- Select the language (foreground and background language)
- PC simulation
- Transfer configuration to the control via the RS-232 interface
- Create the installation disk

2. Online help

It consists of

- Configuring Syntax (PS)
- Development Kit (EU)
- NC variable addresses

3. Microsoft Visual C Workbench

Only the compiler and editor are used.

In addition, you require the following software:

- DOS 6.x
- Windows 3.11
- Microsoft Visual C⁺⁺ 1.5x

3.2 MMC 100/HMI Embedded

You can configure the MMC 100/PCU20 using the Installation Kit and Screen Kit tools.

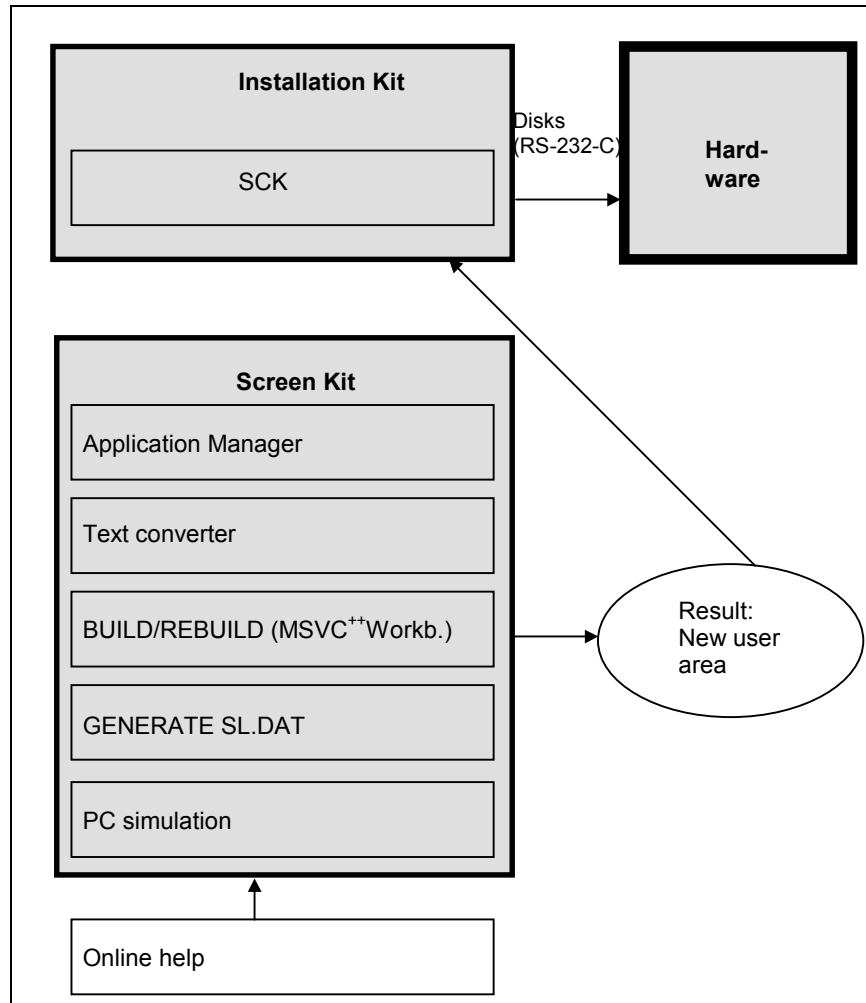


Fig. 3-2 Tools for MMC 100/HMI Embedded

1. Screen Kit

With this you can

- Copy applications using the "Application Manager" (only applications you have created yourself)
- Create new applications
- Create multilingual texts, e.g. using Microsoft Visual C++
- Convert texts using the "the text converter"
- Compile the user area created with Microsoft Visual C++ ("BUILD" or "REBUILD")

- Link the user areas ("Generate SL.DAT")
- Call PC simulation
- Call online help
It consists of
 - Configuring Syntax (PS)
 - Development Kit (EU)
 - NC variable addresses

2. **Installation Kit** with the SCK program
With this you can

- Manage projects
- Assign applications to softkeys
- Change the machine configuration
- Select the language (foreground and background language)
- Transfer the configuration to the control via the RS 232-C interface
- Create disks
 - System disk
 - Application disk
 - Text disk

3. **Microsoft Visual C++ Workbench**

Only the compiler and editor are used.

In addition, you require the following software:

- DOS 6.x
- Windows 3.11 or Windows '95
- Microsoft Visual C++ 1.5x

All functions of the Screen Kit are accessed in the "TOOLS" menu under Microsoft Visual C++:

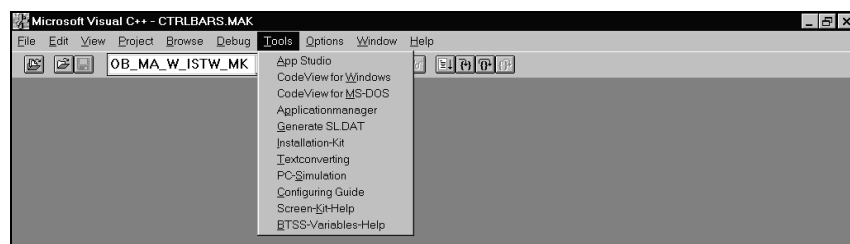


Fig. 3-3 TOOLS menu in the Microsoft Visual C++ Workbench



Configuration Files

4

4.1	General.....	PSE/4-22
4.2	OP 030	PSE/4-25
4.3	MMC 100/HMI Embedded	PSE/4-26

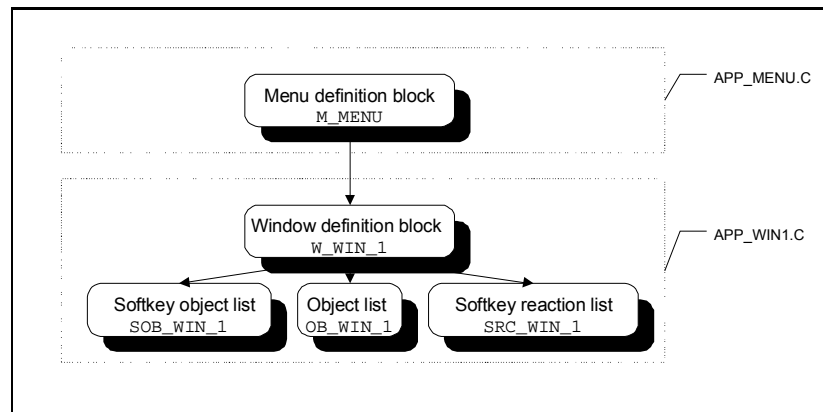
4.1 General

There are 4 categories of configuration files

- *.H Header files
- *.C Source files
- *.MAK Make files (call for compilation)
- *.TXT Text files

Important files

- <Applikat.-Name>.MAK (in the "OBJ_C800" directory)
is the application makefile (management file)
- APP_MENU.C (in the "SRC" directory)
is a sample program. It contains a menu definition block as a starting point in the application.
- <Name>.C (in the "SRC" directory)
can contain window definition blocks and all the lists required for the window.



- AP_L_DIR.H (in the "H" directory)
is a header file. It contains all external cross references to the lists used in the application.
- <Applikat.-Name>.TXT (in the "TEXT" directory)
Text files in various languages
(e.g.: d=German, g=English, e=Spanish, i=Italian).
They contain all the texts required for the application.

APP_MENU.C**Example:**

```
#include "proj.h"
#include "mwl_app.h"
#include "OEM_1.h"
BEGIN_MENU (M_MENU_1)
    M_CLEAR_BACKGROUND, /* clears menu area if menu will be closed */
    0, /* always '0' */
    W_WIN_1, /* id of starting window */
    X_M_INI, Y_M_INI, /* starting position of menu area */
    WIDTH_M_INI, HEIGHT_M_INI, /* width and height of menu area */
    BK_CL_FCOL, /* background color */
    NULL, /* no OPEN_LIST */
    NULL /* no CLOSE_LIST */
END_MENU (M_MENU_1)
```

APP_WIN1.C**Example:**

```
#include "proj.h"
#include "mwl_app.h"
#include "OEM_1.h"
BEGIN_OBJECT_LIST (OB_WIN_1)
    /* clearing and repainting window rectangle */
    RECTANGLE (100, /* id */
               BEGIN_X, BEGIN_Y, /* x-, y-position */
               WIDTH_M_INI, HEIGHT_M_INI, /* width, height of
               rectangle */
               FILLED, /* filled with W_FCOL */
               W_FCOL, /* color of window */
               0xff) /* style of border */
    /* painting window header */
    RECTANGLE (110, BEGIN_X, BEGIN_Y, WIDTH_M_INI, HEADER,
               FILLED, W_HL_FCOL, 0xff)
    /* printing window header text */
    TEXT (120, /* id */
          HEADER_X, HEADER_Y, /* x-, y-position */
          T_APP_WIN_1_HEADER, /* textname */
          CS_SMALL, /* character set */
          0, /* no attributes */
          W_HL_TCOL) /* header-text color */
END_OBJECT_LIST (OB_WIN_1)
```

APP_L_DIR.H

Example:

```
/* APP_MENU.C */  
EXTERN_MENU (M_MENU_1)          /* First entry, never change !!! */  
/* APP_WIN1.C */  
EXTERN_WINDOW (W_WIN_1)  
EXTERN_OBJECT_LIST (OB_WIN_1)  
EXTERN_SOFTKEY_OBJECT_LIST (SOB_WIN_1)
```

OEM_1.TXT

Example:

```
T_APP                          "OEM_01"  
T_APP_WIN_1_HEADER            "My first window"
```


4.2 OP 030

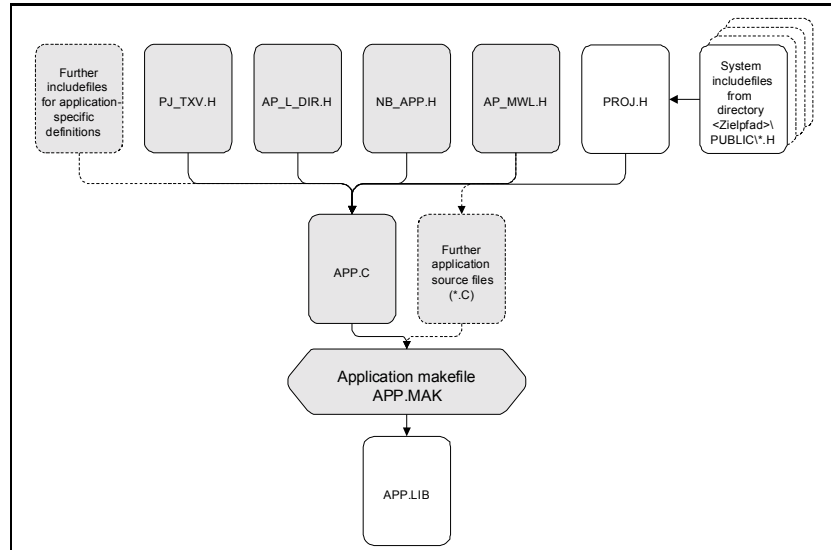


Fig. 4-1 Configuration files for OP 030

Shaded gray: Files to be changed for configuration

4.3 MMC 100/HMI Embedded

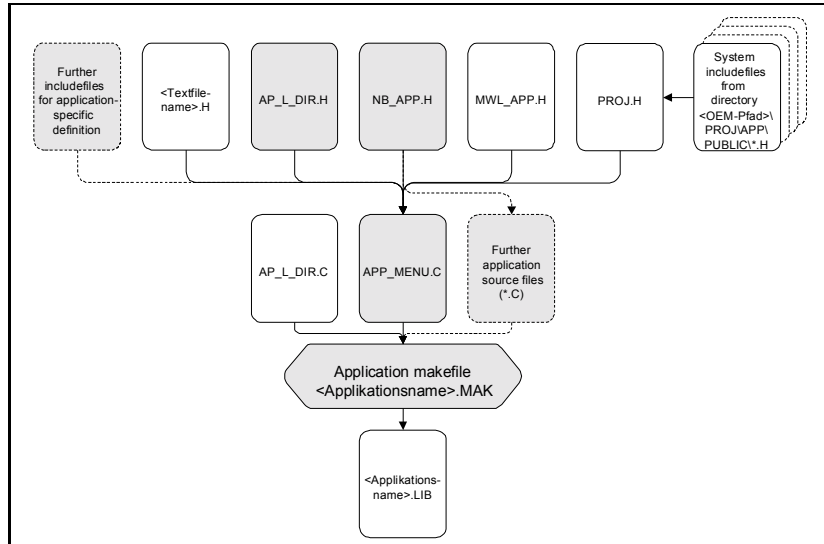


Fig. 4-2 Configuring files for MMC 100/HMI Embedded

Shaded gray: Files to be changed for configuration



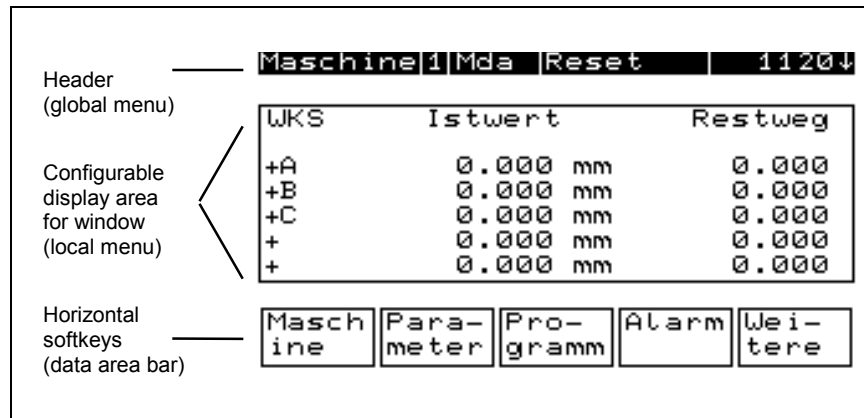
Steps for Configuring the OP 030

5

5.1	User interface	PSE/5-28
5.2	Create application.....	PSE/5-28

5.1 User interface

The user interface is divided into the following areas:



5.2 Create application

Create text for application

An application can be created under the PROJ softkey.

1. Open application in Microsoft Visual C++ (MSVC)
2. Create texts

Convert texts

- a) New application
 1. In the DOS box, select the L:\PROJAPP\TEXT directory
 2. Call "NMAKE.EXE"
- b) Standard application:
 1. In the DOS box, select the L:\PROJ\TEXT directory
 2. Call "NMAKE.EXE"

Note:

The standard application must be converted at least once and, of course, once again if further changes are made.

- Compile application**
- a) New application
 - Either
 - 1. In the DOS box select the L:\PROJ\APP\OBJ_C800 directory
 - 2. Call "NMAKE.EXE APP.MAK"
 - or
 - "Build" or "Rebuild All" in MSVC Workbench

 - b) Standard application:
 - 1. In the DOS box select the L:\PROJ\STD\OBJ_C800 directory
 - 2. Call "NMAKE.EXE STD.MAK"
 - Note:
The standard application must be compiled at least once and, of course, once again if the application source files are changed.
- Link application**
- The project binary file has to be created from the application and standard configuration if at least one source file or text source file has been changed.
- 1. In the DOS box, select the L:\PROJ\DAT directory
 - 2. Call "NMAKE.EXE"
- Test application in simulation**
- a) Offline test
 - 1. In the DOS box, select the L:\BIN directory
 - 2. Call "OP030.EXE"
 - b) Online test via MPI link
 - 1. In the DOS box, select the OP030BIN directory
 - 2. Start the MPI driver by calling COM030.BAT
 - 3. In the L:\BIN directory, copy the BT0_NCK.INI file into the BT0.INI directory.
 - 4. Call "OP030.EXE"
 - 5. End the test (<CTRL> + <X>)

Note

When the user changes from online test mode back to offline test mode, the BT0_SIM.INI file must be copied into BT0.INI again.

**Transfer application
to hardware**

In the DOS box, select the L:\ project drive.

1. Call "INSTALL.BAT"
2. You can select the following components for transfer:
 - System Configuration Files
 - Alarm Texts
 - User Interface Language
 - Application Program
 - System Software
3. Start transfer

Note

The transfer procedure is considerably quicker under MS-DOS than it is under Windows.

Create disks

Two installation disks are created. They contain:

- OP 030 basic system
 - Configuration
 - Binary language files for 5 languages
 - Alarm text source files in 5 languages
 - System configuration files
 - Setup/Install routines
 - MKALARM.BAT for the conversion of alarm texts
1. In the DOS box, select the L:\ directory
 2. Call "MAKEDISK.BAT"



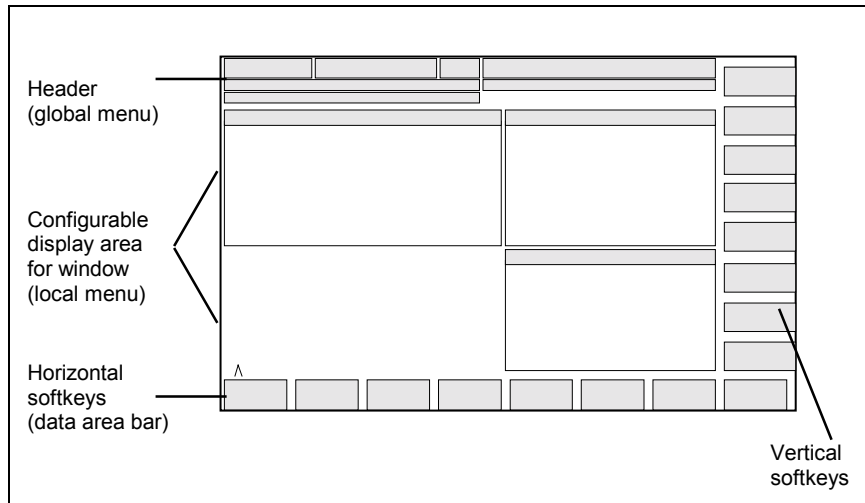
Sequence for Configuring the MMC 100/HMI Embedded

6

6.1	User interface	PSE/6-32
6.2	Create an application.....	PSE/6-33

6.1 User interface

The user interface is divided into the following areas:



6.2 Create an application

Create/copy application	Microsoft Visual C++ Workbench <ol style="list-style-type: none">1. TOOLS menu2. "Application Manager"3. Application New or Application Copy
Add to existing project	Microsoft Visual C++ Workbench <ol style="list-style-type: none">1. TOOLS menu2. "Installation Kit"3. Project Management4. Select project5. Assign softkey6. Define configuration (Create simulation environment)
Configure new project	Microsoft Visual C++ Workbench <ol style="list-style-type: none">1. TOOLS menu2. "Installation Kit"3. Project Management4. New project5. Assign softkey6. Define configuration (Create simulation environment)
Create text for application	Microsoft Visual C++ Workbench <ol style="list-style-type: none">1. FILE menu2. "open"3. Open text file4. Create texts
Convert texts	Microsoft Visual C++ Workbench <ol style="list-style-type: none">1. TOOLS menu2. "Text conversion"3. Set options for conversion

- Compile application** Microsoft Visual C++ Workbench
1. "Project" menu
 2. Build or Rebuild project
 - <Build> only compiles changed application source files again
 - <Rebuild All> compiles all application source files, irrespective of whether they have been changed or not
- Link application** Microsoft Visual C++ Workbench
1. TOOLS menu
 2. "Generate SL.DAT"
- Test application in simulation** Microsoft Visual C++ Workbench
1. TOOLS menu
 2. Start "PC Simulation"
- Transfer application to hardware** Microsoft Visual C++ Workbench
1. TOOLS menu
 2. "Installation Kit"
 - Direct route:
"Install System Software via RS-232-C" menu
 - Indirect route:
via disk
- Create disks** Microsoft Visual C++ Workbench
1. TOOLS menu
 2. "Installation Kit"
 - System disk (with basic system)
 - Application disk
 - Text disk



Compiler Error Messages

7

7.1	Errors.....	PSE/7-36
7.2	Warnings.....	PSE/7-37

7.1 Errors

error C2370 redefinition; different storage class

Description In an application source file a certain ID (here: 110) has been used two or more times for the same macro (here: RECTANGLE).

Elimination

- Use of another ID for one of the two duplicated macros.
- In principle, one ID can be used for several macros. Here it must be ensured that the macros are of a different type.

Example of an error message:

```
d:\mmc0_35.10\proj\app\test\src\app_win1.c(27) : error C2370:  
'rect_110' : redefinition; different storage class
```

Example of a cause:

```
RECTANGLE (110, BEGIN_X, BEGIN_Y,  
           WIDTH_M_INI, HEIGHT_M_INI, FILLED, W_FCOL, 0xff)  
RECTANGLE (110, BEGIN_X, BEGIN_Y,  
           WIDTH_M_INI, HEADER, FILLED, W_HL_FCOL, 0xff)
```

error C2065 undeclared identifier

Description A symbolic text name is specified in a macro (in this case: T_APP_WIN_1_HEADER), which was either not defined in the text source file or was possibly written incorrectly, or was not converted.

Elimination When the symbolic name has not yet been entered in the text file, then add the entry to the file.

When the symbolic name has been written incorrectly, then correct the entries in the application source file.

When the text source file has not been converted, it must be converted before the application is compiled, in as far as a change has been made in the application text files.

Example of an error message:

```
d:\mmc0_35.10\proj\app\test\src\app_win1.c(30) : error C2065:  
'T_APP_HEADER_1' : undeclared identifier
```

generally followed by further error messages.

Example of a cause:

```
TEXT (120, HEADER_X, HEADER_Y, T_APP_WIN_1_HEADER, CS_SMALL, 0,  
     W_HL_TCOL)
```

fatal error C1083 cannot open include file: '...': No such file or directory

Description	The include file to be linked has not been found.
Elimination	<ul style="list-style-type: none"> • Check the path settings in MSVC Workbench: <Options><Directories>[Include Files Path:] • If the header file to be linked is a text include file, then the texts must be converted again. <p>Example of an error message:</p> <pre>d:\mmc0_35.10\proj\app\test\src\app_win1.c(12) : fatal error C1083: Cannot open include file: 'test_01.h': No such file or directory</pre>

7.2 Warnings

warning C4309 'initializing': truncation of constant value

Description	An ID outside the range of [0; 65535] has been used in a macro (in this case: 70000).
Elimination	Use of an ID that lies within the range of [0; 65535].
	<p>Example of an error message:</p> <pre>d:\mmc0_35.10\proj\app\test\src\app_win1.c(45) : warning C4309: 'initializing' : truncation of constant value</pre> <p>Example of a cause:</p> <pre>O_FIELD (70000, X_AXIS_NAME, Y_OUT+0*Y_DIST, 5, W_O_TCOL, W_O_FCOL, CS_SMALL, TEXT_DOUBLE_ZOOMED, 10, P_C_SMA_name, 1, 0, 0, CON_STRING, 0, 0, 0)</pre>

warning C4035 no return value

Description	The name of a macro has been written incorrectly (in this case, correctly: RC_DRAW_SOFTKEY).
Elimination	Correction of the name of the incorrectly written macros.
	Example of an error message:

```
d:\mmc0_35.10\proj\app\test\src\app_win1.c(76) : warning C4035:  
'RC_DROW_SOFTKEY' : no return value
```

generally followed by further error messages.

Example of a cause:

```
RC_DROW_SOFTKEY (510, EVENT_CODE_LESS, 0, KEY_F8_V, 2,  
NOT_PRESSED)
```

warning C4003 not enough actual parameters for macro '...'

1st possibility:

Example of a cause:

```
BEGIN_OPEN_LIST (OP_WIN_1)  
    RC_OPEN_EVENT_LIST (700, EV_WIN_1)  
END_OPEN_LIST (OP_WIN_1)
```

Description

In this example, the incorrect action/reaction element identification (in this case correctly: AC_) has been used for a certain macro.

Elimination

Check of which list is currently being used and use of the appropriate action/reaction list identification:

- AC_...
 OPEN_LIST, CLOSE_LIST, ACTION_LIST
- RC_...
 SOFTKEY_REACTION_LIST, REACTION_LIST

2nd possibility:

Example of a cause:

```
TEXT (150, X_AXIS_UNIT, Y_OUT+0*Y_DIST, T_APP_MM, CS_SMALL,  
TEXT_DOUBLE_ZOOMED)
```

Description

In this example, a parameter has been omitted (in this case: the text color) for the macro.

Elimination

Compare the number of parameters in the configuration and in the online help Configuration Guide. If they are not the same, correct the parameterization of the macros

Example of an error message:

```
d:\mmc0_35.10\proj\app\test\src\app_win1.c(123) : warning C4003:  
not enough actual parameters for macro 'RC_OPEN_EVENT_LIST'
```

warning C4002 too many actual parameters for macro '...'

See "warning C4003: not enough actual parameters for macro '...'"



Examples

8

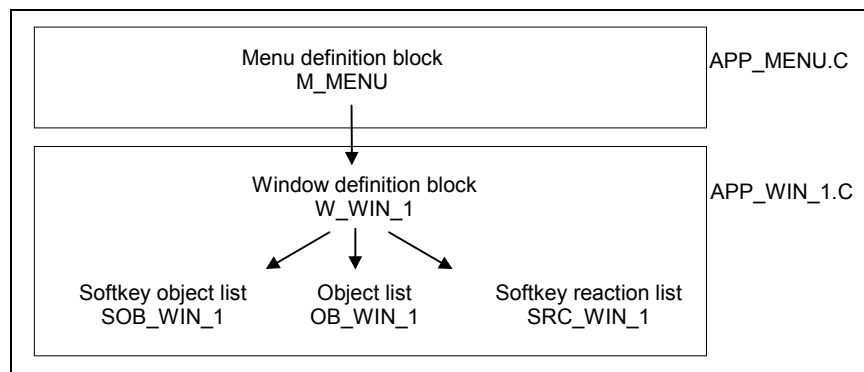
8.1	Change an existing window	PSE/8-40
8.1.1	MMC 100/HMI	PSE/8-40
8.1.2	OP 030	PSE/8-45
8.2	Add window and open using softkey	PSE/8-46
8.2.1	MMC 100/HMI	PSE/8-46
8.2.2	OP 030	PSE/8-51
8.3	Input and output data.....	PSE/8-53
8.4	Results and reactions	PSE/8-58

8.1 Change an existing window

8.1.1 MMC 100/HMI

- Task**
1. Replace the header text of one window in the application by another text.
 2. Give a horizontal softkey the designation "Go to Win2".

Overview



Notes

- The configuration texts can be found in the files ..\PROJ\APP\OEM_1\TEXT\D,G,...\OEM_1.TXT
- When the texts have been changed, it is always necessary to carry out a text converter run and for the application to be compiled.
- The softkey designations can be found in the softkey object list. Configuration Guide: SOFTKEY

SOLUTION

Starting point

Application: OEM_1

APP_MENU.C

```

#include "proj.h"
#include "mwl_app.h"
#include "OEM_1.h"
BEGIN_MENU (M_MENU_1)
    M_CLEAR_BACKGROUND,          /* clears menu area if
                                menu will be closed */
    0,                            /* always '0' */
    W_WIN_1,                      /* id of starting window */

```



```

X_M_INI, Y_M_INI,          /* starting position of menu area */
WIDTH_M_INI, HEIGHT_M_INI, /* width and height of menu area */
BK_CL_FCOL,                /* background color */
NULL,                      /* no OPEN_LIST */
NULL                       /* no CLOSE_LIST */
END_MENU (M_MENU_1)

```

APP_WIN1.C

```

#include "proj.h"
#include "mwl_app.h"
#include "OEM_1.h"
BEGIN_OBJECT_LIST (OB_WIN_1)
    /* clearing and repainting window rectangle */
    RECTANGLE (100,          /* id */
               BEGIN_X, BEGIN_Y, /* x-, y-position */
               WIDTH_M_INI, HEIGHT_M_INI, /* width, height of
               rectangle */
               FILLED,      /* filled with W_FCOL */
               W_FCOL,     /* color of window */
               0xff)       /* style of border */
    /* painting window header */
    RECTANGLE (110, BEGIN_X, BEGIN_Y, WIDTH_M_INI, HEADER,
               FILLED, W_HL_FCOL, 0xff)
    /* printing window header text */
    TEXT (120,          /* id */
          HEADER_X, HEADER_Y, /* x-, y-position */
          T_APP_WIN_1_HEADER, /* textname */
          CS_SMALL,      /* character set */
          0,             /* no attributes */
          W_HL_TCOL)    /* header-text color */
END_OBJECT_LIST (OB_WIN_1)
BEGIN_SOFTKEY_OBJECT_LIST (SOB_WIN_1)
    /* horizontal softkeys */
    SOFTKEY (200,        /* id */
             0,         /* softkeytext */
             KEY_F1,    /* softkeycode */
             2,         /* softkeylines */
             NOT_PRESSED) /* softkeystatus */
    SOFTKEY (210, 0, KEY_F2, 2, NOT_PRESSED)
    SOFTKEY (220, 0, KEY_F3, 2, NOT_PRESSED)
    SOFTKEY (230, 0, KEY_F4, 2, NOT_PRESSED)
    SOFTKEY (240, 0, KEY_F5, 2, NOT_PRESSED)
    SOFTKEY (250, 0, KEY_F6, 2, NOT_PRESSED)
    SOFTKEY (260, 0, KEY_F7, 2, NOT_PRESSED)
    SOFTKEY (270, 0, KEY_F8, 2, NOT_PRESSED)
    /* vertical softkeys */
    SOFTKEY (280, 0, KEY_F1_V, 2, NOT_PRESSED)

```

```

SOFTKEY (290, 0, KEY_F2_V, 2, NOT_PRESSED)
SOFTKEY (300, 0, KEY_F3_V, 2, NOT_PRESSED)
SOFTKEY (310, 0, KEY_F4_V, 2, NOT_PRESSED)
SOFTKEY (320, 0, KEY_F5_V, 2, NOT_PRESSED)
SOFTKEY (330, 0, KEY_F6_V, 2, NOT_PRESSED)
SOFTKEY (340, 0, KEY_F7_V, 2, NOT_PRESSED)
SOFTKEY (350, 0, KEY_F8_V, 2, NOT_PRESSED)
END_SOFTKEY_OBJECT_LIST (SOB_WIN_1)
BEGIN_WINDOW (W_WIN_1)
    0, /* no attribute */
    0, 0, /* rel. window starting
           position to menu */
    WIDTH_M_INI, HEIGHT_M_INI, /* width and height of window
                                area */
    W_BCOL, /* window border color */
    W_FCOL, /* window background color */
    NULL, /* no channel group */
    NULL, /* no OPEN_LIST */
    NULL, /* no CLOSE _LIST */
    OBJECT_LIST_PTR (OB_WIN_1), /* activate OBJECT_LIST
                                'OB_WIN_1' */
    NULL, /* no REACTION_LIST */
    SOFTKEY_OBJECT_LIST_PTR (SOB_WIN_1), /* activatesOFTKEY_OBJECT_
                                           LIST 'SOB_WIN_1' */
    NULL /* no SOFTKEY_REACTION_
           LIST */
END_WINDOW (W_WIN_1)
OEM_1.TXT:
T_APP "OEM_1"
T_APP_WIN_1_HEADER "My first window"
AP_L_DIR.H:
/* APP_MENU.C */
EXTERN_MENU (M_MENU_1) /* First entry, never change
                        !!!!! */
/* APP_WIN1.C */
EXTERN_WINDOW (W_WIN_1)
EXTERN_OBJECT_LIST (OB_WIN_1)
EXTERN_SOFTKEY_OBJECT_LIST (SOB_WIN_1)

```

Change header text of window

1. Open the OEM_1.TXT file in the Microsoft Visual C++ Workbench in the relevant language directory.
2. Replace the entry after "T_APP_WIN_1_Header":
"My first window"
with any text.
3. Initiate the text converter run so that the change will be applied.

Give softkey designation

1. In the "APP_WIN1.C" file, change the softkey designation in the softkey object list:
SOFTKEY(200,0,KEY_F1,2,NOT_PRESSED) in e.g.
SOFTKEY(200,T_SK1H,KEY_F1,2,NOT_PRESSED)
2. Assign a text in the relevant text file OEM_1.txt to the symbolic softkey text name: e.g. T_SK1H "Go to Win2".
3. Initiate the text converter run, compile (everything), link

**Result:
Source file
OEM_2****Application: OEM_2****APP_MENU.C**

See APP_MENU.C from the OEM_1 application

APP_WIN1.C

```
#include "proj.h"
#include "mwl_app.h"
#include "OEM_2.h"

BEGIN_OBJECT_LIST (OB_WIN_1)
    /* clearing and repainting window rectangle */
    RECTANGLE (100,                /* id */
               BEGIN_X, BEGIN_Y,  /* x-, y-position */
               WIDTH_M_INI, HEIGHT_M_INI,
               /* width, height of
               rectangle */
               FILLED,             /* filled with W_FCOL */
               W_FCOL,            /* color of window */
               0xff)              /* style of border */

    /* painting window header */
    RECTANGLE (110, BEGIN_X, BEGIN_Y, WIDTH_M_INI, HEADER, FILLED,
              W_HL_FCOL, 0xff)

    /* printing window header text */
    TEXT (120,                /* id */
          HEADER_X, HEADER_Y, /* x-, y-position */
          T_APP_WIN_1_HEADER, /* textname */
          CS_SMALL,           /* character set */
          0,                  /* no attributes */
          W_HL_TCOL)         /* header-text color */

END_OBJECT_LIST (OB_WIN_1)

BEGIN_SOFTKEY_OBJECT_LIST (SOB_WIN_1)
    /* horizontal softkeys */
    SOFTKEY (200,                /* id */
             T_SK1H_WIN1,       /* softkeytext */
             KEY_F1,            /* softkeycode */
             2,                  /* softkeylines */
             NOT_PRESSED)       /* softkeystatus */

    SOFTKEY (210, 0, KEY_F2, 2, NOT_PRESSED)
    SOFTKEY (220, 0, KEY_F3, 2, NOT_PRESSED)
    SOFTKEY (230, 0, KEY_F4, 2, NOT_PRESSED)
    SOFTKEY (240, 0, KEY_F5, 2, NOT_PRESSED)
    SOFTKEY (250, 0, KEY_F6, 2, NOT_PRESSED)
    SOFTKEY (260, 0, KEY_F7, 2, NOT_PRESSED)
    SOFTKEY (270, 0, KEY_F8, 2, NOT_PRESSED)

    /* vertical softkeys */
```

```
SFTKEY (280, 0, KEY_F1_V, 2, NOT_PRESSED)
SFTKEY (290, 0, KEY_F2_V, 2, NOT_PRESSED)
SFTKEY (300, 0, KEY_F3_V, 2, NOT_PRESSED)
SFTKEY (310, 0, KEY_F4_V, 2, NOT_PRESSED)
SFTKEY (320, 0, KEY_F5_V, 2, NOT_PRESSED)
SFTKEY (330, 0, KEY_F6_V, 2, NOT_PRESSED)
SFTKEY (340, 0, KEY_F7_V, 2, NOT_PRESSED)
SFTKEY (350, 0, KEY_F8_V, 2, NOT_PRESSED)
END_SFTKEY_OBJECT_LIST (SOB_WIN_1)
BEGIN_WINDOW (W_WIN_1)
    0,                                /* no attribute */
    0, 0,                              /* rel. window starting position to
                                     menu */
    WIDTH_M_INI, HEIGHT_M_INI,         /* width and height of window
                                     area */
    W_BCOL,                             /* window border color */
    W_FCOL,                             /* window background color */
    NULL,                               /* no channel group */
    NULL,                               /* no OPEN_LIST */
    NULL,                               /* no CLOSE_LIST */
    OBJECT_LIST_PTR (OB_WIN_1),        /* activate OBJECT_LIST
                                     'OB_WIN_1' */
    NULL,                               /* no REACTION_LIST */
    SFTKEY_OBJECT_LIST_PTR (SOB_WIN_1), /* activate SFTKEY_OBJECT_LIST
                                     'SOB_WIN_1' */
    NULL                                /* no SFTKEY_REACTION_LIST */
END_WINDOW (W_WIN_1)
```

OEM_2.TXT:

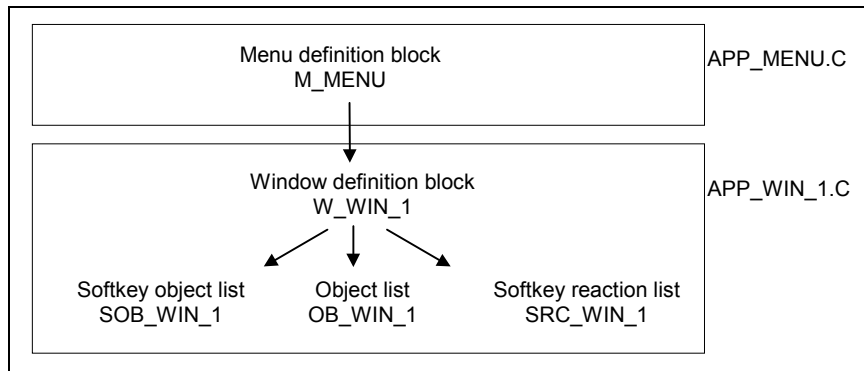
```
T_APP                                "OEM_2"
T_APP_WIN_1_HEADER                   "OEM: My first window"
T_SK1H_WIN1                          "go to%n Win 2"
```

AP_L_DIR.H:

```
/* APP_MENU.C */
EXTERN_MENU (M_MENU_1)
/* APP_WIN1.C */
EXTERN_WINDOW (W_WIN_1)
EXTERN_OBJECT_LIST (OB_WIN_1)
EXTERN_SFTKEY_OBJECT_LIST (SOB_WIN_1)
```

8.1.2 OP 030

- Task**
1. Replace the header text of one window in the application by another text.
 2. Give a horizontal softkey the designation "Go to Win2".

Overview**Notes**

- The configuration texts can be found in the files
..\PROJ\APP\OEM_1\TEXT\D,G,...\OEM_1.TXT
- When the texts have been changed, it is always necessary to carry out a text converter run and for the application to be compiled.
- The softkey designations can be found in the softkey object list.
Configuration Guide: SOFTKEY

SOLUTION**Change header text of window**

Microsoft Visual C++ Workbench.

1. Open OEM_1.txt file in the relevant language directory.
2. Replace the entry "T_APP_WIN_1HEADER" with "....".
3. Initiate the text converter run so that the change will be applied.

Give softkey designation

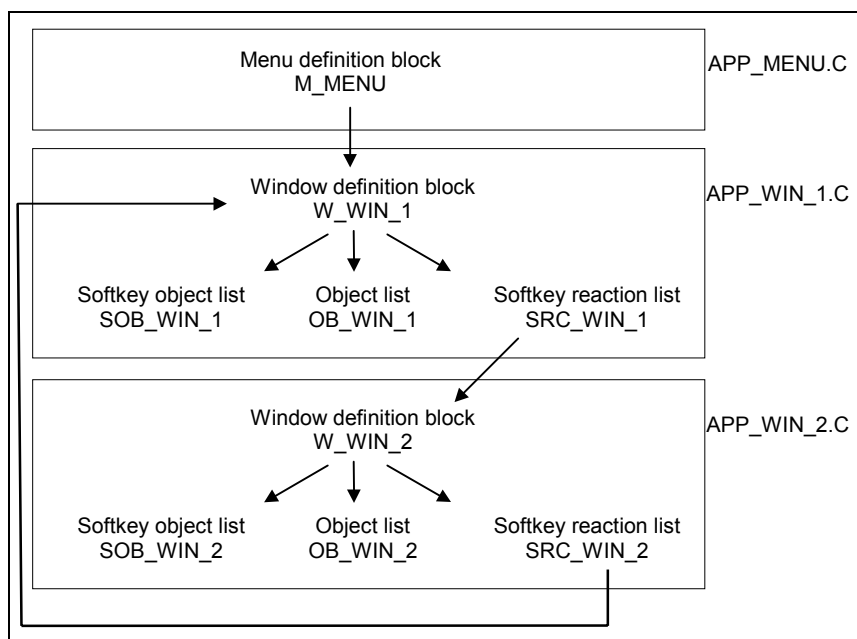
1. In the "APP_WIN.C" file, change the softkey designation in the softkey object list
SOFTKEY(200,0,KEY_F1,2,NOT_PRESSED) in e.g.
SOFTKEY(200,T_SK1H,KEY_F1,2,NOT_PRESSED)
2. Assign a text in the relevant text file OEM_1.txt to the symbolic softkey text name:
e.g. T_SK1H "Go to Win2".
3. Initiate the text converter run, compile (everything), link.

8.2 Add window and open using softkey

8.2.1 MMC 100/HMI

- Task**
1. Add a window to the application.
 2. Press the softkey to close the window currently open and open the other one in each case.

Overview



Operating steps

- Link an additional application source file into the MSVC Workbench application makefile: <Project> <Edit...> menu.
- Each list to which an application is added must be entered in the application list directory AP_L_DIR.H with an appropriate external reference.
- Close the windows:
Configuration Guide: CLOSE_WINDOW
- Open the windows:
Configuration Guide: OPEN_WINDOW
- A window can be activated (once) via a reference in the menu definition block. It is closed again automatically when the menu definition block is closed.

- Order in which lists are processed when opening or closing a window:
 1. Processing of the open list
 2. Processing of the the object list
 3. Activation of the reaction list
 4. Activation of the softkey object list and softkey reaction list

Steps 1 and 2 are interchanged by setting the attribute
W_OPEN_AFTER_OBJ in the window definition block.

- If further routines are called in an action, reaction or softkey reaction list in which OPEN_WINDOW is called, these will be executed before the new reaction or softkey reaction list becomes active.

SOLUTION

Starting point: OEM_2 application

Add window

MSVC Workbench

1. Open OEM_2.MAK application.
2. Open APP_WIN1.C file and save under another name (e.g. APP_WIN2.C).
3. Select <Project> <Edit> menu.
Incorporate the newly created application source file in the current project.
4. Change list name in the APP_WIN2.C file from _WIN_1" to "_WIN_2".
5. In the AP_L_DIR.H file, add entries for the new lists of the APP_WIN2.C file.
6. Add to the text file the entry for the softkey designation and the header line of the new window.

Open the window using the softkey

7. One softkey reaction list must be configured for each window in order to be able to open the second window by activating the softkey.
8. Link the macros CLOSE_WINDOW and OPEN_WINDOW into the softkey reaction lists.

Result: Source_file OEM_3

Application: OEM_3

APP_MENU:

See APP_MENU.C of the OEM_1 application

APP_WIN1.C

```
#include "proj.h"
#include "mwl_app.h"
#include "OEM_3.h"
EXTERN_SOFTKEY_OBJECT_LIST (SOB_CLEAR_SOFTKEYS)
```

```

BEGIN_OBJECT_LIST (OB_WIN_1)
    /* clearing and repainting window rectangle */
    RECTANGLE (100,                /* id */
              BEGIN_X, BEGIN_Y,    /* x-, y-position */
              WIDTH_M_INI, HEIGHT_M_INI,
                                      /* width, height of
                                      rectangle */
              FILLED,                /* filled with W_FCOL */
              W_FCOL,                /* color of window */
              0xff)                  /* style of border */
    /* painting window header */
    RECTANGLE (110, BEGIN_X, BEGIN_Y, WIDTH_M_INI, HEADER,
              FILLED, W_HL_FCOL, 0xff)
    /* printing window header text */
    TEXT (120,                      /* id */
         HEADER_X, HEADER_Y,        /* x-, y-position */
         T_APP_WIN_1_HEADER,        /* textname */
         CS_SMALL,                  /* character set */
         0,                          /* no attributes */
         W_HL_TCOL)                 /* header-text color */
END_OBJECT_LIST (OB_WIN_1)

BEGIN_SOFTKEY_OBJECT_LIST (SOB_WIN_1)
    MACRO (200, 0, 0, SOFTKEY_OBJECT_LIST_PTR
          (SOB_CLEAR_SOFTKEYS))
        SOFTKEY (210,                /* id */
                T_SK1H_WIN1,         /* softkeytext */
                KEY_F1,              /* softkeycode */
                2,                    /* softkeylines */
                NOT_PRESSED)         /* softkeystatus */
END_SOFTKEY_OBJECT_LIST (SOB_WIN_1)

BEGIN_SOFTKEY_REACTION_LIST (SRC_WIN_1)
    RC_CLOSE_WINDOW (300,            /* id */
                    KEY_F1,          /* softkeycode */
                    W_WIN_1,         /* name of window */
                    LOCAL,           /* menutyp */
                    1)               /* clears window */
    RC_OPEN_WINDOW (310,             /* id */
                   KEY_F1,           /* softkeycode */
                   W_WIN_2,          /* name of window */
                   LOCAL)            /* menutyp */
END_SOFTKEY_REACTION_LIST (SRC_WIN_1)

BEGIN_WINDOW (W_WIN_1)
    0,                                /* no attribute */
    0, 0,                             /* rel. window starting position
    to menu */
    WIDTH_M_INI, HEIGHT_M_INI,
                                      /* width and height of window
                                      area */
    W_BCOL,                            /* window border color */

```



```

W_FCOL,                /* window background color */
NULL,                  /* no channel group */
NULL,                  /* no OPEN_LIST */
NULL,                  /* no CLOSE_LIST */
OBJECT_LIST_PTR (OB_WIN_1),
                        /* activate OBJECT_LIST
                        'OB_WIN_1' */
NULL,                  /* no REACTION_LIST */
SOFTKEY_OBJECT_LIST_PTR (SOB_WIN_1),
                        /* activate SOFTKEY_OBJECT_LIST
                        'SOB_WIN_1' */
SOFTKEY_REACTION_LIST_PTR (SRC_WIN_1)
                        /* activates SOFTKEY_REACTION_LIST
                        'SRC_WIN_1' */

END_WINDOW (W_WIN_1)

```

APP_WIN2.C:

```

#include "proj.h"
#include "mwl_app.h"
#include "OEM_3.h"
#include "nb_app.h"
EXTERN_SOFTKEY_OBJECT_LIST (SOB_CLEAR_SOFTKEYS)
BEGIN_OBJECT_LIST (OB_WIN_2)
    RECTANGLE (1100, BEGIN_X, BEGIN_Y, WIDTH_M_INI,
                HEIGHT_M_INI, FILLED, W_FCOL, 0xff)
    RECTANGLE (1110, BEGIN_X, BEGIN_Y, WIDTH_M_INI, HEADER,
                FILLED, W_HL_FCOL, 0xff)
    TEXT (1120, HEADER_X, HEADER_Y, T_APP_WIN_2_HEADER,
          CS_SMALL, 0, W_HL_TCOL)
END_OBJECT_LIST (OB_WIN_2)
BEGIN_SOFTKEY_OBJECT_LIST (SOB_WIN_2)
    MACRO (1200, 0, 0, SOFTKEY_OBJECT_LIST_PTR
           (SOB_CLEAR_SOFTKEYS))
        SOFTKEY (1210,          /* id */
                 T_SK1H_WIN2,  /* softkeytext */
                 KEY_F1,       /* softkeycode */
                 2,            /* softkeylines */
                 NOT_PRESSED) /* softkeystatus */
    END_SOFTKEY_OBJECT_LIST (SOB_WIN_2)
BEGIN_SOFTKEY_REACTION_LIST (SRC_WIN_2)
    RC_CLOSE_WINDOW (1300, /* id */
                     KEY_F1, /* softkeycode */
                     W_WIN_2, /* name of window */
                     LOCAL, /* menutyp */
                     1) /* clears window */
    RC_OPEN_WINDOW (1310, /* id */
                    KEY_F1, /* softkeycode */
                    W_WIN_1, /* name of window */
                    LOCAL) /* menutyp */
END_SOFTKEY_REACTION_LIST (SRC_WIN_2)
BEGIN_WINDOW (W_WIN_2)

```

```
0,  
0, 0,  
WIDTH_M_INI, HEIGHT_M_INI,  
W_BCOL,  
W_FCOL,  
NULL,  
NULL,  
NULL,  
OBJECT_LIST_PTR (OB_WIN_2),  
NULL,  
SOFTKEY_OBJECT_LIST_PTR (SOB_WIN_2), /* activates  
SOFTKEY_OBJECT_LIST 'SOB_WIN_2' */  
SOFTKEY_REACTION_LIST_PTR (SRC_WIN_2)/* activates  
SOFTKEY_REACTION_LIST 'SRC_WIN_2' */  
END_WINDOW (W_WIN_2)
```

OEM_3.TXT:

```
T_APP                "OEM_3"  
T_APP_WIN_1_HEADER  "OEM: My first window"  
T_SK1H_WIN1         "go to%n Win 2"  
T_APP_WIN_2_HEADER  "OEM: My second window"  
T_SK1H_WIN2         "go to%n Win 1"
```

AP_L_DIR.H:

```
/* APP_MENU.C */  
EXTERN_MENU (M_MENU_1  
/* APP_WIN1.C */  
EXTERN_WINDOW (W_WIN_1)  
EXTERN_OBJECT_LIST (OB_WIN_1)  
EXTERN_SOFTKEY_OBJECT_LIST (SOB_WIN_1)  
EXTERN_SOFTKEY_REACTION_LIST (SRC_WIN_1)  
/* APP_WIN2.C */  
EXTERN_WINDOW (W_WIN_2)  
EXTERN_OBJECT_LIST (OB_WIN_2)  
EXTERN_SOFTKEY_OBJECT_LIST (SOB_WIN_2)  
EXTERN_SOFTKEY_REACTION_LIST (SRC_WIN_2)  
/* APP_FNCT.C */  
EXTERN_SOFTKEY_OBJECT_LIST (SOB_CLEAR_SOFTKEYS)
```

APP_FNCT.C:

```
#include "proj.h"  
#include "mwl_app.h"  
#include "OEM_3.h"  
#include "nb_app.h"  
BEGIN_SOFTKEY_OBJECT_LIST (SOB_CLEAR_SOFTKEYS)  
    SOFTKEY (10000, 0, KEY_F1, 2, NOT_PRESSED)  
    SOFTKEY (10001, 0, KEY_F2, 2, NOT_PRESSED)
```

```

SOFTKEY (10002, 0, KEY_F3, 2, NOT_PRESSED)
SOFTKEY (10003, 0, KEY_F4, 2, NOT_PRESSED)
SOFTKEY (10004, 0, KEY_F5, 2, NOT_PRESSED)
SOFTKEY (10005, 0, KEY_F6, 2, NOT_PRESSED)
SOFTKEY (10006, 0, KEY_F7, 2, NOT_PRESSED)
SOFTKEY (10007, 0, KEY_F8, 2, NOT_PRESSED)
SOFTKEY (10008, 0, KEY_F1_V, 2, NOT_PRESSED)
SOFTKEY (10009, 0, KEY_F2_V, 2, NOT_PRESSED)
SOFTKEY (10010, 0, KEY_F3_V, 2, NOT_PRESSED)
SOFTKEY (10011, 0, KEY_F4_V, 2, NOT_PRESSED)
SOFTKEY (10012, 0, KEY_F5_V, 2, NOT_PRESSED)
SOFTKEY (10013, 0, KEY_F6_V, 2, NOT_PRESSED)
SOFTKEY (10014, 0, KEY_F7_V, 2, NOT_PRESSED)
SOFTKEY (10015, 0, KEY_F8_V, 2, NOT_PRESSED)
END_SOFTKEY_OBJECT_LIST (SOB_CLEAR_SOFTKEYS)

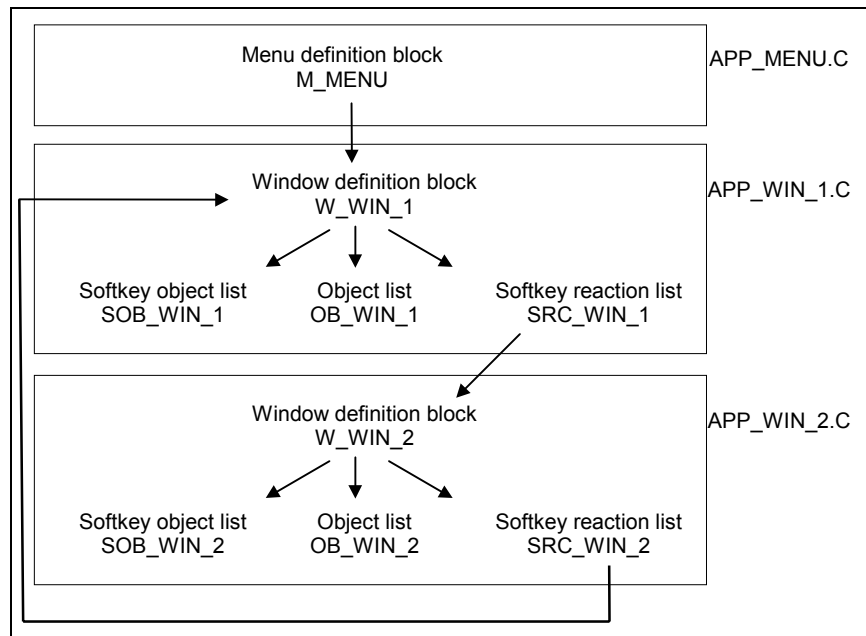
```

8.2.2 OP 030

Task

1. Add a window to the application.
2. Press the softkey to close the window currently open and open the other one in each case.

Overview



Operating steps

- Link an additional application source file into the application makefile:
MSVC Workbench: <Project> <Edit...> menu
- Each list to which an application is added must be entered in the application list directory AP_L_DIR.H with an appropriate external reference.
- Close the windows:
Configuration Guide: CLOSE_WINDOW
- Open the windows:
Configuration Guide: OPEN_WINDOW
- A window can be activated (once) via a reference in the menu definition block. It is closed again automatically when the menu definition block is closed.

- Order in which lists are processed when opening or closing a window:
 1. Processing of the open list
 2. Processing of the the object list
 3. Activation of the reaction list
 4. Activation of the softkey object list and softkey reaction list

Steps 1 and 2 are interchanged by setting the attribute W_OPEN_AFTER_OBJ in the window definition block.
- If further routines are called in an action, reaction or softkey reaction list in which OPEN_WINDOW is called, these will be executed before the new reaction or softkey reaction list becomes active.

SOLUTION

Add window

MSVC Workbench

1. Open OEM_2.MAK application.
2. Open APP_WIN1.C and save under another name (e.g. APP_WIN2.C).
3. Select <Project> <Edit> menu.
Incorporate the newly created application source file in the current project.
4. Change list name in the APP_WIN2.C file from "_WIN_1" to "_WIN_2".
5. In the AP_L_DIR.H file, add entries for the new lists of the APP_WIN2.C file.
6. Add to the text file the entry for the softkey designation and the header line of the new window.

Open the window using the softkey

7. One softkey reaction list must be configured for each window in order to be able to open the second window by activating the softkey.
8. Link the macros CLOSE_WINDOW and OPEN_WINDOW into the softkey reaction lists.

8.3 Input and output data

Task

1. The axis name, the current value and the unit of the first axis in the current channel is to be shown in the first window.
The font is to be assigned double height and width.
2. In the second line, configure an input and output field that shows the current value of R10 and is followed by the comment "R Parameter 10".
3. For every entry into the first window, R10 is to be initialized with the value "123.456".
4. If the cursor is used in the input/output field for displaying R10, a text with information is to be output in the status line.



Operating steps

- Dynamic output fields are configured in the object list: Configuring Guide: O_FIELD, IO_FIELD
- The symbolic names of the NCK addresses are taken from "BTSS Variables Help".
- The position of the outputs is to be defined in their own header file (e.g. APP_INCL.H). This file must be included in every application source file in which these defines are used!
- The following parameters for data conversion are used:
CON_STRING: Axis name
CON_DECIMAL (F_DOUBLE): Axis position, R parameter
(see Configuring Guide: Data conversion)
- Data initializations are generally configured in open lists.

- Set a double value: Configuration Guide: SET_DOUBLE

Graphical display elements	Display variants for NC/PLC/MMC data
Dot PIXEL	Input/output field O_FIELD, I_FIELD, IO_FIELD
Line LINE	Selection field CHECK_FIELD
Rectangle RECTANGLE	Inverse field INVERSE_FIELD
Circle (MMC100/HMI Embedded only) CIRCLE	Pictogram field POLYMARKER
Ellipse (MMC100/HMI Embedded only) ELLIPSE	Action field ACTION_FIELD
Arc (MMC100/HMI Embedded only) ARC	Scrollbar DEF_SCROLLBAR
Fill pattern for graphical display elements DEF_PATTERN	
Text TEXT	

SOLUTION

Starting point for the OEM_3 application

1. Open the OEM_3.MAK application in the MSVC Workbench
2. Open the APP_WIN1.C file.
3. Configure fields:
 - Output field (O_FIELD) which displays the axis names.
 - Output field (O_FIELD) which displays the current axis value.
 - Text output (TEXT) for displaying the unit in "mm".
 - Input/output field (IO_FIELD) which displays the current value of R10.
 - Text output (TEXT) to display the "R parameter" name.
4. Store texts required in the application source files.

Result:
Source file
OEM_4

Application: OEM_4

APP_MENU.C:

See OEM_1 application

APP_WIN1.C:

```
#include "proj.h"
#include "mwl_app.h"
#include "OEM_4.h"
#include "app_incl.h"
EXTERN_SOFTKEY_OBJECT_LIST (SOB_CLEAR_SOFTKEYS)
BEGIN_OBJECT_LIST (OB_WIN_1)
    /* clearing and repainting window rectangle */
    RECTANGLE (100, /* id */
               BEGIN_X, BEGIN_Y, /* x-, y-position */
```

```

        WIDTH_M_INI, HEIGHT_M_INI,
                                /* width, height of
                                rectangle */
        FILLED,                    /* filled with W_FCOL */
        W_FCOL,                    /* color of window */
        0xff)                      /* style of border */
/* painting window header */
RECTANGLE (110, BEGIN_X, BEGIN_Y, WIDTH_M_INI, HEADER,
           FILLED, W_HL_FCOL, 0xff)
/* printing window header text */
TEXT (120,                        /* id */
      HEADER_X, HEADER_Y,         /* x-, y-position */
      T_APP_WIN_1_HEADER,        /* textname */
      CS_SMALL,                  /* character set */
      0,                          /* no attributes */
      W_HL_TCOL)                 /* header-text color */

/* show name of first axis of actual channel */
O_FIELD (130,                      /* id */
         X_AXIS_NAME, Y_OUT+0*Y_DIST, 5,
                                /* x-, y-position, width in
                                characters */
         W_O_TCOL, W_O_FCOL,      /* text color, background
                                color */
         CS_SMALL,                /* character set */
         TEXT_DOUBLE_ZOOMED,     /* double height and double
                                width */
         10,                      /* data refresh */
         P_C_SMA_name, 1, 0, 0,   /* adr. of first axis name in
                                actual channel */
         CON_STRING, 0, 0, 0)     /* show data as string */
/* show value of first axis of actual channel */
O_FIELD (140,
         X_AXIS_VALUE, Y_OUT+0*Y_DIST, 10,
         W_O_TCOL, W_O_FCOL,
         CS_SMALL,
         TEXT_DOUBLE_ZOOMED,
         1,
         P_C_SMA_actToolBasePos, 1, 0, 0, /* adr. of first
                                axis val. in act. chan. */
         CON_DECIMAL, F_DOUBLE, 3, 0) /* show data as
                                decimal */
/* show unit of first axis of actual channel */
TEXT (150, X_AXIS_UNIT, Y_OUT+0*Y_DIST, T_APP_MM
      0, CS_SMALL, TEXT_DOUBLE_ZOOMED, W_TCOL)
/* show actual value of r-parameter */
IO_FIELD (160,                      /* id */
          X_R_PAR_VALUE, Y_OUT+1*Y_DIST, 16,
                                /* x-, y-position, width in
                                characters */
          W_IO_TCOL, W_IO_FCOL, /* text and background
                                color */

```

```

        CS_SMALL,                /* character set */
        TEXT_DOUBLE_ZOOMED,     /* double height and double
                                width */
        0, 0, 0, 0,            /* move cursor to field nr.
                                [] if arrow keys pressed r,
                                l, d, u */
        0,                      /* access class */
        1,                      /* data refresh */
        0,                      /* show text, if cursor is on
                                this field */
        P_C_RP_rpa, (10+1), 0, 0,
                                /* address of
                                r-parameter 10 */
        CON_DECIMAL, F_DOUBLE, 8, 0)
                                /* show data as decimal */
    TEXT (170, X_R_PAR_NAME, Y_OUT+1*Y_DIST, T_APP_R_PAR_10,
         CS_SMALL, TEXT_DOUBLE_ZOOMED, W_TCOL)
END_OBJECT_LIST (OB_WIN_1)
BEGIN_SOFTKEY_OBJECT_LIST (SOB_WIN_1)
    MACRO (200, 0, 0, SOFTKEY_OBJECT_LIST_PTR
         (SOB_CLEAR_SOFTKEYS))
        SOFTKEY (210,          /* id */
                 T_SK1H_WIN1, /* softkeytext */
                 KEY_F1,      /* softkeycode */
                 2,          /* softkeylines */
                 NOT_PRESSED) /* softkeystatus */
    END_SOFTKEY_OBJECT_LIST (SOB_WIN_1)
BEGIN_SOFTKEY_REACTION_LIST (SRC_WIN_1)
    RC_CLOSE_WINDOW (300,    /* id */
                    KEY_F1, /* softkeycode */
                    W_WIN_1, /* name of window */
                    LOCAL,  /* menutyp */
                    1)      /* clears window */
    RC_OPEN_WINDOW (310,    /* id */
                   KEY_F1, /* softkeycode */
                   W_WIN_2, /* name of window */
                   LOCAL)  /* menutyp */
END_SOFTKEY_REACTION_LIST (SRC_WIN_1)
BEGIN_WINDOW (W_WIN_1)
    0,                /* no attribute */
    0, 0,            /* rel. window starting
                    position to menu */
    WIDTH_M_INI, HEIGHT_M_INI, /* width and height of window
                                area */
    W_BCOL,          /* window border color */
    W_FCOL,          /* window background color */
    NULL,            /* no channel group */
    NULL,            /* no OPEN_LIST */
    NULL,            /* no CLOSE_LIST */
    OBJECT_LIST_PTR (OB_WIN_1), /* activate OBJECT_LIST
                                'OB_WIN_1' */
    NULL,            /* no REACTION_LIST */

```



```

SOFTKEY_OBJECT_LIST_PTR (SOB_WIN_1),
                        /* activateSOFTKEY_OBJECT_
                        LIST 'SOB_WIN_1' */

SOFTKEY_REACTION_LIST_PTR (SRC_WIN_1)
                        /* activates SOFTKEY_
                        REACTION_LIST 'SRC_WIN_1' */

END_WINDOW (W_WIN_1)

```

APP_WIN2.C:

see APP_WIN2.C of the OEM_3 application

APP_FNCT.C:

see APP_FNCT.C of the OEM_3 application

AP_L_DIR.H:

```

/* APP_MENU.C */
EXTERN_MENU (M_MENU_1)

/* APP_WIN1.C */
EXTERN_WINDOW (W_WIN_1)
EXTERN_OBJECT_LIST (OB_WIN_1)
EXTERN_SOFTKEY_OBJECT_LIST (SOB_WIN_1)
EXTERN_SOFTKEY_REACTION_LIST (SRC_WIN_1)

/* APP_WIN2.C */
EXTERN_WINDOW (W_WIN_2)
EXTERN_OBJECT_LIST (OB_WIN_2)
EXTERN_SOFTKEY_OBJECT_LIST (SOB_WIN_2)
EXTERN_SOFTKEY_REACTION_LIST (SRC_WIN_2)

/* APP_FNCT.C */
EXTERN_SOFTKEY_OBJECT_LIST (SOB_CLEAR_SOFTKEYS)

```

APP_INCL.H:

```

#define X_AXIS_NAME      10
#define X_AXIS_VALUE    100
#define X_AXIS_UNIT     300
#define Y_OUT           40
#define Y_DIST          40
#define X_R_PAR_VALUE  10
#define X_R_PAR_NAME   300

```

OEM_4.TXT:

```

T_APP                "OEM_4"
T_APP_WIN_1_HEADER  "OEM: My first window"
T_SK1H_WIN1         "go to%n Win 2"
T_APP_WIN_2_HEADER  "OEM: My second window"
T_SK1H_WIN2         "go to%n Win 1"
T_APP_MM            "mm"
T_APP_R_PAR_10      "R-Parameter 10"

```

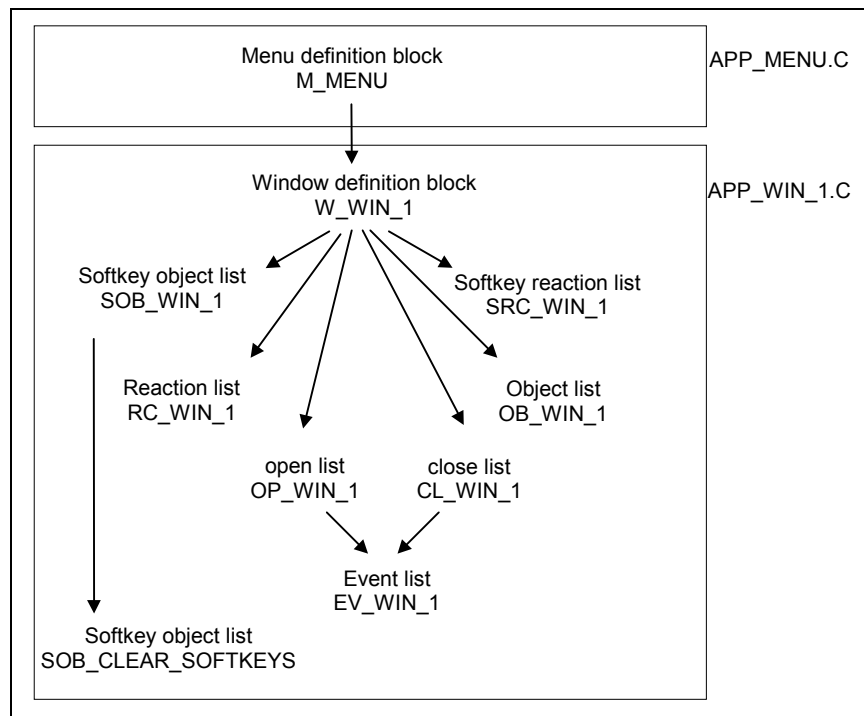
8.4 Results and reactions

Task

1. An event list (EVENT_LIST) is to be configured which monitors the value of R10 as follows:
Event code 1, if the change is from $R10 \leq 100$ to > 100 .
Event code 2, if the change is from $R10 > 100$ to ≤ 100 .
2. The following is to be configured as the reaction to the event codes:
Event code 1:
A softkey is to be designated "Reset R10".
If this softkey is pressed, the value of R10 is set to 0.
Event code 2:
The softkey "Reset R10" is to be deleted again.
The functionality of the softkey is to be disabled again.
3. Event code values must be in the range [10000; 19999].
They can be declared as follows, e.g. in the APP_INCL.H file:

```
#define EVENT_CODE_LESS 10000
#define EVENT_CODE_GREATER 10001
```

Overview



Operating steps

- An event list (EVENT_LIST) must be activated explicitly with the macro OPEN_EVENT_LIST and deactivated again with CLOSE_EVENT_LIST. Generally, an open list (OPEN_LIST) is used for activation and a close list (CLOSE_LIST) is used for deactivation.
Configuration Guide: OPEN_EVENT_LIST, CLOSE_EVENT_LIST, EVENT_LIST, WATCH_EVENT
- The reactions are configured in a reaction list (REACTION_LIST). Activation is by means of a reference in the window definition block.
- Give softkey a designation:
Configuration Guide: DRAW_SOFTKEY
- Set a double variable:
Configuration Guide: SET_DOUBLE
- Activation/deactivation of a function (e.g. setting a double variable):
Configuration Guide: SKIP_IF

SOLUTION

Starting point: Source file OEM_4

1. Open the OEM_4.MAK application in the MSVC Workbench.
2. Open the APP_WIN1.C application source files.
3. Configure event list (EVENT_LIST), reaction list (REACTION_LIST), open list (OPEN_LIST) and close list (CLOSE_LIST).
4. Enter external references in the application list directory AP_L_DIR.H and enter references in the window definition block.
5. Open the event list in the open list using the macro OPEN_EVENT_LIST. Close the event list in the close list using the macro CLOSE_EVENT_LIST.
6. Configure WATCH_EVENT that reacts to the change $R10 > 100$.
Configure WATCH_EVENT that reacts to the change $R10 \leq 100$.
7. Define both event codes in an application include file (e.g. APP_INCL.H).
8. Configure the softkey designation, irrespective of the respective event code, using the macro DRAW_SOFTKEY in the reaction list.
9. In order to enable the softkey function "Reset R10" as a function of the status of R10, before the SET_DOUBLE a step function SKIP_IF must be configured which, depending on the content of R10, may or may not jump the next command(s).

Result:
Source file
OEM_5

Application: OEM_5
APP_MENU.C:

see APP_MENU.C of the OEM_5 application

APP_WIN1.C:

```
#include "proj.h"
#include "mwl_app.h"
#include "OEM_5.h"
#include "app_incl.h"
EXTERN_SOFTKEY_OBJECT_LIST (SOB_CLEAR_SOFTKEYS)
BEGIN_OBJECT_LIST (OB_WIN_1)
    /* clearing and repainting window rectangle */
    RECTANGLE (100,          /* id */
               BEGIN_X, BEGIN_Y, /* x-, y-position */
               WIDTH_M_INI, HEIGHT_M_INI,
               /* width, height of
               rectangle */
               FILLED,      /* filled with W_FCOL */
               W_FCOL,     /* color of window */
               0xff)       /* style of border */
    /* painting window header */
    RECTANGLE (110, BEGIN_X, BEGIN_Y, WIDTH_M_INI, HEADER,
              FILLED, W_HL_FCOL, 0xff)
    /* printing window header text */
    TEXT (120,          /* id */
          HEADER_X, HEADER_Y, /* x-, y-position */
          T_APP_WIN_1_HEADER, /* textname */
          CS_SMALL,      /* character set */
          0,             /* no attributes */
          W_HL_TCOL)     /* header-text color */
    /* show name of first axis of actual channel */
    O_FIELD (130,      /* id */
            X_AXIS_NAME, Y_OUT+0*Y_DIST, 5,
            /* x-, y-position, width in
            characters */
            W_O_TCOL, W_O_FCOL, /* text color, background
            color */
            CS_SMALL, /* character set */
            TEXT_DOUBLE_ZOOMED, /* double height and double
            width */
            10, /* data refresh */
            P_C_SMA_name, 1, 0, 0,
            /* adr. of first axis name
            in act. chan.*/
            CON_STRING, 0, 0, 0) /* show data as string */
    /* show value of first axis of actual channel */
```

```

O_FIELD (140,
        X_AXIS_VALUE, Y_OUT+0*Y_DIST, 10,
        W_O_TCOL, W_O_FCOL,
        CS_SMALL,
        TEXT_DOUBLE_ZOOMED,
        1,
        P_C_SMA_actToolBasePos, 1, 0, 0,
        /* adr. of first axis value
        in act. channel */
        CON_DECIMAL, F_DOUBLE, 3, 0)
        /* show data as decimal */
/* show unit of first axis of actual channel */
TEXT (150, X_AXIS_UNIT, Y_OUT+0*Y_DIST, T_APP_MM, CS_SMALL,
TEXT_DOUBLE_ZOOMED, W_TCOL)
/* show actual value of r-parameter */
IO_FIELD (160, /* id */
        X_R_PAR_VALUE, Y_OUT+1*Y_DIST, 16,
        /* x-, y-position, width in
        characters */
        W_IO_TCOL, W_IO_FCOL, /* text and background
        color */
        CS_SMALL, /* character set */
        TEXT_DOUBLE_ZOOMED, /* double height and double
        width */
        0, 0, 0, 0, /* move cursor to field nr.
        [] if arrow keys pressed r,
        l, d, u */
        0, /* access class */
        1, /* data refresh */
        0, /* show text, if cursor is on
        this field */
        P_C_RP_rpa, (10+1), 0, 0,
        /* address of r-parameter
        10 */
        CON_DECIMAL, F_DOUBLE, 8, 0)
        /* show data as decimal */
TEXT (170, X_R_PAR_NAME, Y_OUT+1*Y_DIST, T_APP_R_PAR_10,
CS_SMALL,
TEXT_DOUBLE_ZOOMED, W_TCOL)
END_OBJECT_LIST (OB_WIN_1)
BEGIN_REACTION_LIST (RC_WIN_1)
RC_DRAW_SOFTKEY (500, EVENT_CODE_GREATER, T_SK8V_WIN_1,
KEY_F8_V, 2, NOT_PRESSED)
RC_DRAW_SOFTKEY (510, EVENT_CODE_LESS, 0, KEY_F8_V, 2,
NOT_PRESSED)
END_REACTION_LIST (RC_WIN_1)

```

```

BEGIN_EVENT_LIST (EV_WIN_1)
  /* examines the value R10 */
  /* if this value becomes less equal than 100
     the event-code EVENT_CODE_LESS (10000) is sent */
  WATCH_EVENT (400, 1, EVENT_CODE_LESS, /* ID, refresh, event
                                         code to send */
              FALSE_TO_TRUE|WATCH_LESS_EQUAL,
              /* watch attribute */
              F_DOUBLE, /* data typ */
              100.0, /* compare value */
              P_C_RP_rpa, (10+1), 0, 0, 0)
              /* address of watch
              value */

  /* examines the value in notebook NB_REAC_EVENT_EXAMPLE (nr.
     201) */
  /* if this value becomes greater than 100
     the event-code EVENT_CODE_GREATER (10001) is sent */
  WATCH_EVENT (410, 1, EVENT_CODE_GREATER,
              FALSE_TO_TRUE|WATCH_GREATER,
              F_DOUBLE, 100.0, P_C_RP_rpa, (10+1), 0, 0, 0)
END_EVENT_LIST (EV_WIN_1)

BEGIN_SOFTKEY_OBJECT_LIST (SOB_WIN_1)
  MACRO (200, 0, 0, SOFTKEY_OBJECT_LIST_PTR
        (SOB_CLEAR_SOFTKEYS))
  SOFTKEY (210, /* id */
          T_SKIH_WIN1, /* softkeytext */
          KEY_F1, /* softkeycode */
          2, /* softkeylines */
          NOT_PRESSED) /* softkeystatus */
END_SOFTKEY_OBJECT_LIST (SOB_WIN_1)

BEGIN_SOFTKEY_REACTION_LIST (SRC_WIN_1)
  RC_CLOSE_WINDOW (300, /* id */
                  KEY_F1, /* softkeycode */
                  W_WIN_1, /* name of window */
                  LOCAL, /* menutyp */
                  1) /* clears window */
  RC_OPEN_WINDOW (310, /* id */
                 KEY_F1, /* softkeycode */
                 W_WIN_2, /* name of window */
                 LOCAL) /* menutyp */
  RC_C_D_SKIP_IF (320, KEY_F8_V, 1, 100.0, GREATER_EQUAL,
                 P_C_RP_rpa, (10+1), 0, 0)
  RC_SET_DOUBLE (330, KEY_F8_V, 0.0, P_C_RP_rpa, (10+1), 0, 0)
END_SOFTKEY_REACTION_LIST (SRC_WIN_1)

```

```

BEGIN_OPEN_LIST (OP_WIN_1)
    AC_OPEN_EVENT_LIST (700, EV_WIN_1)
END_OPEN_LIST (OP_WIN_1)
BEGIN_CLOSE_LIST (CL_WIN_1)
    AC_CLOSE_EVENT_LIST (800, EV_WIN_1)
END_CLOSE_LIST (CL_WIN_1)
BEGIN_WINDOW (W_WIN_1)
    0,                                /* no attribute */
    0, 0,                              /* rel. window starting position to
                                     menu */
    WIDTH_M_INI, HEIGHT_M_INI,        /* width and height of window
                                     area */
    W_BCOL,                             /* window border color */
    W_FCOL,                             /* window background color */
    NULL,                               /* no channel group */
    OPEN_LIST_PTR (OP_WIN_1),          /* activate OPEN_LIST 'OP_WIN_1' */
    CLOSE_LIST_PTR (CL_WIN_1),        /* activate CLOSE_LIST 'CL_WIN_1' */
    OBJECT_LIST_PTR (OB_WIN_1),       /* activate OBJECT_LIST
                                     'OB_WIN_1' */
    REACTION_LIST_PTR (RC_WIN_1),     /* activate REACTION_LIST
                                     'RC_WIN_1' */
    SOFTKEY_OBJECT_LIST_PTR (SOB_WIN_1), /* activate SOFTKEY_OBJECT_LIST
                                     'SOB_WIN_1' */
    SOFTKEY_REACTION_LIST_PTR (SRC_WIN_1) /* activate SOFTKEY_REACTION_LIST
                                     'SRC_WIN_1' */
END_WINDOW (W_WIN_1)

```

APP_WIN2.C:

see APP_WIN2.C of the OEM_3 application

APP_FNCT.C:

see APP_FNCT.C of the OEM_3 application

AP_L_DIR.H:

```

/* APP_MENU.C */
EXTERN_MENU (M_MENU_1)
/* APP_WIN1.C */
EXTERN_WINDOW (W_WIN_1)
EXTERN_OBJECT_LIST (OB_WIN_1)
EXTERN_SOFTKEY_OBJECT_LIST (SOB_WIN_1)
EXTERN_SOFTKEY_REACTION_LIST (SRC_WIN_1)
EXTERN_REACTION_LIST (RC_WIN_1)
EXTERN_EVENT_LIST (EV_WIN_1)
EXTERN_OPEN_LIST (OP_WIN_1)
EXTERN_CLOSE_LIST (CL_WIN_1)
/* APP_WIN2.C */
EXTERN_WINDOW (W_WIN_2)
EXTERN_OBJECT_LIST (OB_WIN_2)
EXTERN_SOFTKEY_OBJECT_LIST (SOB_WIN_2)
EXTERN_SOFTKEY_REACTION_LIST (SRC_WIN_2)
/* APP_FNCT.C */
EXTERN_SOFTKEY_OBJECT_LIST (SOB_CLEAR_SOFTKEYS)

```

APP_INCL.H:

```
#define X_AXIS_NAME          10
#define X_AXIS_VALUE        100
#define X_AXIS_UNIT          300
#define Y_OUT                40
#define Y_DIST              40
#define X_R_PAR_VALUE        10
#define X_R_PAR_NAME        300
/* event codes */
#define EVENT_CODE_LESS     10000
#define EVENT_CODE_GREATER  10001
```

OEM_5.TXT:

```
T_APP                       "OEM_5"
T_APP_WIN_1_HEADER          "OEM: My first window"
T_SK1H_WIN1                 "go to%n Win 2"
T_APP_WIN_2_HEADER          "OEM: My second window"
T_SK1H_WIN2                 "go to%n Win 1"
T_APP_MM                    "mm"
T_APP_R_PAR_10              "R-Parameter 10"
T_SK8V_WIN_1                "Reset%n R10"
```



Tips

9

9.1	General.....	PSE/9-66
-----	--------------	----------

9.1 General

1. All lists assigned to a window should be placed in an applications source file.
2. List identifiers should contain the window name in coded form in their name (readability of the source code).
3. Never make more than a few changes or extensions so that errors can be assigned (traced) more easily to a certain change/extension after a compiler run.



Index

10

A

Actions 1-8
Application user area 1-10

C

Colors 1-11
Compiler error messages 7-35
Compiling 1-10
Configuration files: General 4-22
Configuration lists 1-9
Configuration of OP 030 5-27
Configuration Tools: OP 030 3-18
Configuring data 1-10
Configuring files: MMC 100/HMI 4-26
Configuring language 1-4
Configuring the MMC 100/HMI Embedded 6-31
Configuring tools: MMC 100/HMI 3-19
Create application 5-28

D

Development Kit 1-4
Dialog fields 1-6
Display elements 1-6
Display objects 1-6
Documentation 1-4
Dynamic display elements 1-6

E

Error C1083 7-37
Error C2065 7-36
Error C2370 7-36
Events 1-8
Example: Change existing window 8-40

H

Hardware: Overview 2-15

I

Installation 1-4

L

Languages 1-11
Linking 1-10
List elements 1-9

List entries 1-9
List identifiers 1-9
Lists 1-9

M

Menu 1-7, 1-8
MMC 100/HMI: User interface 6-32
MMC variables 1-6
MMC 100/HMI: Create an application 6-33

N

NCK interface 1-6
Number ranges 1-11

O

Objects 1-6
OP 030 4-25
OP 030: User interface 5-28
OP 030: Configuration 5-27
Operation sequence 1-7
Operator's guide 1-4

R

Reactions 1-8
Requirements 1-4, 2-14

S

Simulation 1-10
Softkey 1-7
Standard user areas 1-10

T

Test 1-10
Text sizes 1-11
Texts 1-11
Tips 9-66

U

User area 1-8

V

Variables 1-6

W

Warning C4002 7-38
Warning C4003 7-38
Warning C4035 7-37
Warning C4309 7-37
Window 1-7
Window sizes 1-11

References

A

General Documentation

- /BU/** SINUMERIK 840D/840Di/810D/802S, C, D
Ordering Information
Catalog NC 60.1
Order number: E86060-K4460-A101-A8-7600
- /ST7/** **SIMATIC**
SIMATIC S7 Programmable Logic Controller
Catalog ST 70
Order number: E86060-K4670-A111-A3
- /Z/** SINUMERIK, SIROTEC, SIMODRIVE
Accessories and Equipment for Special Purpose Machines
Catalog NC Z
Order number: E86060-K4490-A001-A7-7600

Electronic Documentation

- /CD8/** The SINUMERIK System (01.02 Edition)
DOC ON CD
(includes all SINUMERIK 840D/840Di/810D and SIMODRIVE publications)
Order number: 6FC5 298-6CA00-0BG2

User Documentation

/AUK/	SINUMERIK 840D/810D Short Guide AutoTurn Operation Order number: 6FC5 298-4AA30-0BP3	(11.01 Edition)
/AUP/	SINUMERIK 840D/810D AutoTurn Graphic Programming System Operator's Guide Programming / Setting Up Order number: 6FC5 298-4AA40-0BP3	(11.01 Edition)
/BA/	SINUMERIK 840D/810D Operator's Guide Order number: 6FC5 298-6AA00-0BP0	(10.00 Edition)
/BAD/	SINUMERIK 840D/840Di/810D Operator's Guide: HMI Advanced Order number: 6FC5 298-6AF00-0BP1	(11.01 Edition)
/BEM/	SINUMERIK 840D/810D Operator's Guide: HMI Embedded Order number: 6FC5 298-6AC00-0BP1	(11.01 Edition)
/BAE/	SINUMERIK 840D/810D Operator's Guide Unit Operator Panel Order number: 6FC5 298-3AA60-0BP1	(04.96 Edition)
/BAH/	SINUMERIK 840D/840Di/810D Operator's Guide HT 6 (HPU New) Order number: 6FC5 298-0AD60-0BP2	(11.01 Edition)
/BAK/	SINUMERIK 840D/840Di/810D Short Operating Guide Order number: 6FC5 298-6AA10-0BP0	(02.01 Edition)
/BAM/	SINUMERIK 810D/840D Operator's Guide ManualTurn Order number: 6FC5 298-6AD00-0BP0	(10.01 Edition)
/BAS/	SINUMERIK 840D/810D Operator's Guide ShopMill Order number: 6FC5 298-6AD10-0BP0	(10.01 Edition)
/BAT/	SINUMERIK 840D/810D Operator's Guide ShopTurn Order number: 6FC5 298-6AD50-0BP0	(03.01 Edition)
/BAP/	SINUMERIK 840D/840Di/810D Short Guide Handheld Programming Unit Order number: 6FC5 298-5AD20-0BP1	(04.00 Edition)
/BNM/	SINUMERIK 840D/840Di/810D User's Guide Measuring Cycles Order number: 6FC5 298-6AA70-0BP1	(09.01 Edition)

/DA/	SINUMERIK 840D/840Di/810D Diagnostics Guide Order number: 6FC5 298-6AA20-0BP1	(09.01 Edition)
/KAM/	SINUMERIK 840D/810D Short Guide ManualTurn Order number: 6FC5 298-5AD40-0BP0	(04.01 Edition)
/KAS/	SINUMERIK 840D/810D Short Guide ShopMill Order number: 6FC5 298-5AD30-0BP0	(04.01 Edition)
/PG/	SINUMERIK 840D/840Di/810D Programming Guide Fundamentals Order number: 6FC5 298-6AB00-0BP1	(09.01 Edition)
/PGA/	SINUMERIK 840D/840Di/810D Programming Guide Advanced Order number: 6FC5 298-6AB10-0BP1	(09.01 Edition)
/PGK/	SINUMERIK 840D/840Di/810D Short Guide Programming Order number: 6FC5 298-6AB30-0BP1	(02.01 Edition)
/PGM	SINUMERIK 840D/840Di/810D Programming Guide ISO Milling Order number: 6FC5 298-6AC20-0BP1	(10.01 Edition)
/PGT/	SINUMERIK 840D/840Di/810D Programming Guide ISO Turning Order number: 6FC5 298-6AC10-0BP1	(10.01 Edition)
/PGZ/	SINUMERIK 840D/840Di/810D Programming Guide Cycles Order number: 6FC5 298-6AB40-0BP1	(09.01 Edition)
/PI/	PCIN 4.4 Software for Data Transfer to/from MMC Module Order number: 6FX2 060-4AA00-4XB0 (English, French, German) Order from: WK Fürth	
/SYI/	SINUMERIK 840Di System Overview Order number: 6FC5 298-6AE40-0BP0	(02.01 Edition)

Manufacturer/Service Documentation

a) Lists

/LIS/ SINUMERIK 840D/840Di/810D
SIMODRIVE 611D
Lists (09.01 Edition)
Order number: 6FC5 297-6AB70-0BP1

b) Hardware

/BH/ SINUMERIK 840D/840Di/810D
Operator Components Manual (HW) (09.01 Edition)
Order number: 6FC5 297-6AA50-0BP1

/BHA/ SIMODRIVE **Sensor**
Absolute Encoder with PROFIBUS DP
User's Guide (HW) (02.99 Edition)
Order number: 6SN1 197-0AB10-0YP1

/EMV/ SINUMERIK, SIROTEC, SIMODRIVE
EMC Installation Guideline (06.99 Edition)
Planning Guide (HW)
Order number: 6FC5 297-0AD30-0BP1

/PHC/ SINUMERIK 810D
Configuring Manual (HW) (12.01 Edition)
Order number: 6FC5 297-4AD10-0BP1

/PHD/ SINUMERIK 840D
NCU 561.2-573.2 Configuring Manual (HW) (09.01 Edition)
Order number: 6FC5 297-6AC10-0BP1

/PHF/ SINUMERIK FM-NC
NCU 570 Configuring Manual (HW) (04.96 Edition)
Order number: 6FC5 297-3AC00-0BP0

/PMH/ SIMODRIVE **Sensor**
Measuring System for Main Spindle Drives
SIMAG-H Configuring/Installation Guide (HW) (05.99 Edition)
Order number: 6SN1197-0AB30-0BP0

c) Software

/FB1/	SINUMERIK 840D/840Di/810D/FM-NC Description of Functions, Basic Machine (Part 1) (the various manuals are listed below) Order number: 6FC5 297-6AC20-0BP1	(09.01 Edition)
	A2 Various Interface Signals	
	A3 Axis Monitoring, Protection Zones	
	B1 Continuous Path Mode, Exact Stop and Look Ahead	
	B2 Acceleration	
	D1 Diagnostic Tools	
	D2 Interactive Programming	
	F1 Travel to Fixed Stop	
	G2 Velocities, Setpoint/Actual Value Systems, Closed-Loop Control	
	H2 Output of Auxiliary Functions to PLC	
	K1 Mode Group, Channels, Program Operation	
	K2 Coordinate Systems, Axis Types, Axis Configurations, Actual-Value System for Workpiece, External Zero Offset	
	K4 Communication	
	N2 EMERGENCY STOP	
	P1 Transverse Axes	
	P3 Basic PLC Program	
	R1 Reference Point Approach	
	S1 Spindles	
	V1 Feeds	
	W1 Tool Compensation	
/FB2/	SINUMERIK 840D/840Di/810D(CCU2) Description of Functions, Extension Functions (Part 2) including FM-NC: Turning, Stepping Motor (the various manuals are listed below) Order number: 6FC5 297-6AC30-0BP1	(09.01 Edition)
	A4 Digital and Analog NCK I/Os	
	B3 Several Operator Panels and NCUs	
	B4 Operation via PG/PC	
	F3 Remote Diagnostics	
	H1 Jog with/without Handwheel	
	K3 Compensations	
	K5 Mode Groups, Channels, Axis Replacement	
	L1 FM-NC Local Bus	
	M1 Kinematic Transformation	
	M5 Measurements	
	N3 Software Cams, Position Switching Signals	
	N4 Punching and Nibbling	
	P2 Positioning Axes	
	P5 Oscillation	
	R2 Rotary Axes	
	S3 Synchronous Spindles	
	S5 Synchronized Actions (SW 3 and earlier, later /FBSY/)	
	S6 Stepper Motor Control	
	S7 Memory Configuration	
	T1 Indexing Axes	
	W3 Tool Change	
	W4 Grinding	

/FB3/	<p>SINUMERIK 840D/840Di/810D(CCU2) Description of Functions Special Functions (Part 3) (09.01 Edition) (the various manuals are listed below) Order number: 6FC5 297-6AC80-0BP1</p> <p>F2 3-Axis to 5-Axis Transformation G1 Gantry Axes G3 Cycle Times K6 Contour Tunnel Monitoring M3 Coupled Axes and ESR S8 Constant Workpiece Speed for Centerless Grinding T3 Tangential Control TE1 Clearance Control TE2 Analog Axis TE3 Speed/Torque Coupling Master-Slave TE4 Transformation Handling Package TE5 Setpoint Exchange TE6 MCS Coupling TE7 Retrace Support TE8 Unlocked Path-Synchronous Switching Signal Output V2 Preprocessing W5 3D Tool Radius Compensation</p>
/FBA/	<p>SIMODRIVE 611D/SINUMERIK 840D/810D Description of Functions, Drive Functions (09.01 Edition) (the various sections are listed below) Order number: 6SN1 197-0AA80-0BP7</p> <p>DB1 Operational Messages/Alarm Reactions DD1 Diagnostic Functions DD2 Speed Control Loop DE1 Extended Drive Functions DF1 Enable Commands DG1 Encoder Parameterization DM1 Calculation of Motor/Power Section Parameters and Controller Data DS1 Current Control Loop DÜ1 Monitors/Limitations</p>
/FBAN/	<p>SINUMERIK 840D/SIMODRIVE 611 DIGITAL Description of Functions ANA-MODULE (02.00 Edition) Order number: 6SN1 197-0AB80-0BP0</p>
/FBD/	<p>SINUMERIK 840D Description of Functions Digitizing (07.99 Edition) Order number: 6FC5 297-4AC50-0BP0</p> <p>DI1 Start-Up DI2 Scanning with Tactile Sensors (scancad scan) DI3 Scanning with Lasers (scancad laser) DI4 Milling Program Generation (scancad mill)</p>
/FBDN/	<p>CAM Integration DNC NT-2000 Description of Functions System for NC Data Management and Data Distribution (05.00 Edition) Order number: 6FC5 297-5AE50-0BP1</p>
/FBDT/	<p>IT Solutions NC Data Transmission (SinDNC) (03.01 Edition) Description of Functions Order number: 6FC5 297-1AE70-0BP0</p>

/FBFA/	SINUMERIK 840D/840Di/810D Description of Functions ISO Dialects for SINUMERIK Order number: 6FC5 297-6AE10-0BP1	(09.01 Edition)
/FBFE/	SINUMERIK 840D/810D Description of Functions Remote Diagnostics Order number: 6FC5 297-6AF00-0BP1 FE1 Remote Diagnostics FE2 Interrupt-Controlled Email Messaging: @Event	(09.01 Edition)
/FBHLA/	SINUMERIK 840D/SIMODRIVE 611 digital Description of Functions HLA Module Order number: 6SN1 197-0AB60-0BP2	(04.00 Edition)
/FBMA/	SINUMERIK 840D/810D Description of Functions ManualTurn Order number: 6FC5 297-6AD50-0BP0	(10.01 Edition)
/FBO/	SINUMERIK 840D/810D Description of Functions Configuring of OP 030 Operator Interface (the various sections are listed below) Order number: 6FC5 297-6AC40-0BP0 BA Operator's Guide EU Development Environment (Configuring Package) PS Online only: Configuring Syntax (Configuring Package) PSE Introduction to Configuring of Operator Interface IK Screen Kit: Software Update and Configuration CS only online: Configuring Syntax (Configuring Package): is included with the software and available as a pdf.	(09.01 Edition)
/FBP/	SINUMERIK 840D Description of Functions C-PLC Programming Order number: 6FC5 297-3AB60-0BP0	(03.96 Edition)
/FBR/	SINUMERIK 840D/810D Description of Functions SINCOM Computer Link Order number: 6FC5 297-5AD60-0BP0 NFL Interface to Central Production Computer NPL Interface to PLC/NCK	(04.00 Edition)
/FBSI/	SINUMERIK 840D/SIMODRIVE Description of Functions SINUMERIK Safety Integrated Order number: 6FC5 297-6AB80-0BP0	(03.01 Edition)
/FBSP/	SINUMERIK 840D/810D Description of Functions ShopMill Order number: 6FC5 297-6AD80-0BP1	(12.01 Edition)
/FBST/	SIMATIC Description of Functions FM STEPDRIVE/SIMOSTEP Order number: 6SN1 197-0AA70-0YP4	(01.01 Edition)

A References

/FBSY/	SINUMERIK 840D/810D Description of Functions Synchronized Actions for Wood, Glass, Ceramics and Presses Order number: 6FC5 297-6AD40-0BP1	(09.01 Edition)
/FBT/	SINUMERIK 840D/810D Description of Functions ShopTurn Order number: 6FC5 297-6AD70-0BP0	(03.01 Edition)
/FBU/	SIMODRIVE 611 universal Description of Functions Closed-Loop Control Component for Speed Control and Positioning Order number: 6SN1 197-0AB20-0BP4	(08.01 Edition)
/FBW/	SINUMERIK 840D/810D Description of Functions Tool Management Order number: 6FC5 297-6AC60-0BP1	(10.01 Edition)
/HBI/	SINUMERIK 840Di Manual Order number: 6FC5 297-6AE60-0BP0	(09.01 Edition)
/KBU/	SIMODRIVE 611 universal Short Description Closed-Loop Control Component for Speed Control Order number: 6SN1 197-0AB40-0BP3	(05.00 Edition)
/PJE/	SINUMERIK 840D/810D HMI Embedded Configuring Package Description of Functions: Software Update, Configuration, Installation Order number: 6FC5 297-6EA10-0BP0 (the CS Configuring Syntax is included with the software and available as a pdf)	(08.01 Edition)
/PJFE/	SIMODRIVE Planning Guide 1FE1 Synchronous Built-In Motors AC Motors for Main Spindle Drives Order number: 6SN1 197-0AC00-0BP1	(09.01 Edition)
/PJLM/	SIMODRIVE Planning Guide Linear Motors (on request)	(06.01 Edition)
	ALL General Information about Linear Motors 1FN1 1FN1 Three-Phase Linear Motor 1FN3 1FN3 Three-Phase Linear Motor CON Connections Order number: 6SN1 197-0AB70-0BP2	

/PJM/	SIMODRIVE Planning Guide Motors AC Motors for Feed and Main Spindle Drives Order number: 6SN1 197-0AA20-0BP4	(09.00 Edition)
/PJU/	SIMODRIVE 611 Planning Guide Converters Order number: 6SN1 197-0AA00-0BP5	(05.01 Edition)
/POS1/	SIMODRIVE POSMO A Operator's Guide Distributed Positioning Motor on PROFIBUS DP, Order number: 6SN2 197-0AA00-0BP3	(08.01 Edition)
/POS2/	SIMODRIVE POSMO A Installation Instructions (supplied with every POSMO A) Order number: 462 008 0815 00	(12.98 Edition)
/POS3/	SIMODRIVE POSMO SI/CD/CA Operator's Guide Distributed Servo Drive Systems Order number: 6SN2 197-0AA20-0BP1	(08.01 Edition)
/S7H/	SIMATIC S7-30 Manual: Assembly, CPU Data (HW) Reference Manual: Module Data Order number: 6ES7 398-8AA03-8AA0	(10.98 Edition)
/S7HT/	SIMATIC S7-300 STEP7 Manual, Basic Information, V3.1 Order number: 6ES7 810-4CA02-8AA0	(03.97 Edition)
/S7HR/	SIMATIC S7-300 Manual STEP7, Reference Manuals, V3.1 Order number: 6ES7 810-4CA02-8AR0	(03.97 Edition)
/S7S/	SIMATIC S7-300 FM 353 Stepper Drive Positioning Module Order in conjunction with configuring package	(04.97 Edition)
/S7L/	SIMATIC S7-300 FM 354 Servo Drive Positioning Module Order in conjunction with configuring package	(04.97 Edition)
/S7M/	SIMATIC S7-300 FM 357.2 Multi-Axis Module for Servo and Stepper Drives Order in conjunction with configuring package	(01.01 Edition)
/SHM/	SIMODRIVE 611 Manual Single-Axis Positioning Control for MCU 172A Order number: 6SN 1197-4MA00-0BP0	(01.98 Edition)
/SP/	SIMODRIVE 611-A/611-D, SimoPro 3.1 Program for Configuring Machine-Tool Drives Order number: 6SC6 111-6PC00-0AA□ Order from: WK Fürth	

d) Installation and start-up

/IAA/	SIMODRIVE 611A Installation and Start-Up Guide (including description of SIMODRIVE 611D start-up software) Order number: 6SN 1197-0AA60-0BP6	(10.00 Edition)
/IAC/	SINUMERIK 810D Installation and Start-Up Guide (including description of SIMODRIVE 611D start-up software) Order number: 6FC5 297-4AD20-0BP1	(12.01 Edition)
/IAD/	SINUMERIK 840D/SIMODRIVE 611D Installation and Start-Up Guide (including description of SIMODRIVE 611D start-up software) Order number: 6FC5 297-6AB10-0BP1	(09.01 Edition)
/IAF/	SINUMERIK FM-NC Installation and Start-Up Guide Order number: 6FC5 297-3AB00-0BP1	(07.00 Edition)
/IAM/	SINUMERIK 840D/840Di/810D HMI/MMC Installation and Start-Up Guide Order number: 6FC5 297-6AE20-0BP1	(11.01 Edition)
	AE1 Updates/Options	
	BE1 Expand the operator interface	
	HE1 Online Help	
	IM2 Start-Up HMI Embedded	
	IM4 Start-Up HMI Advanced (PCU 50)	
	TX1 Setting Foreign Language Texts	



To
 SIEMENS AG
 A&D MC BMS
 P.O. Box 3180

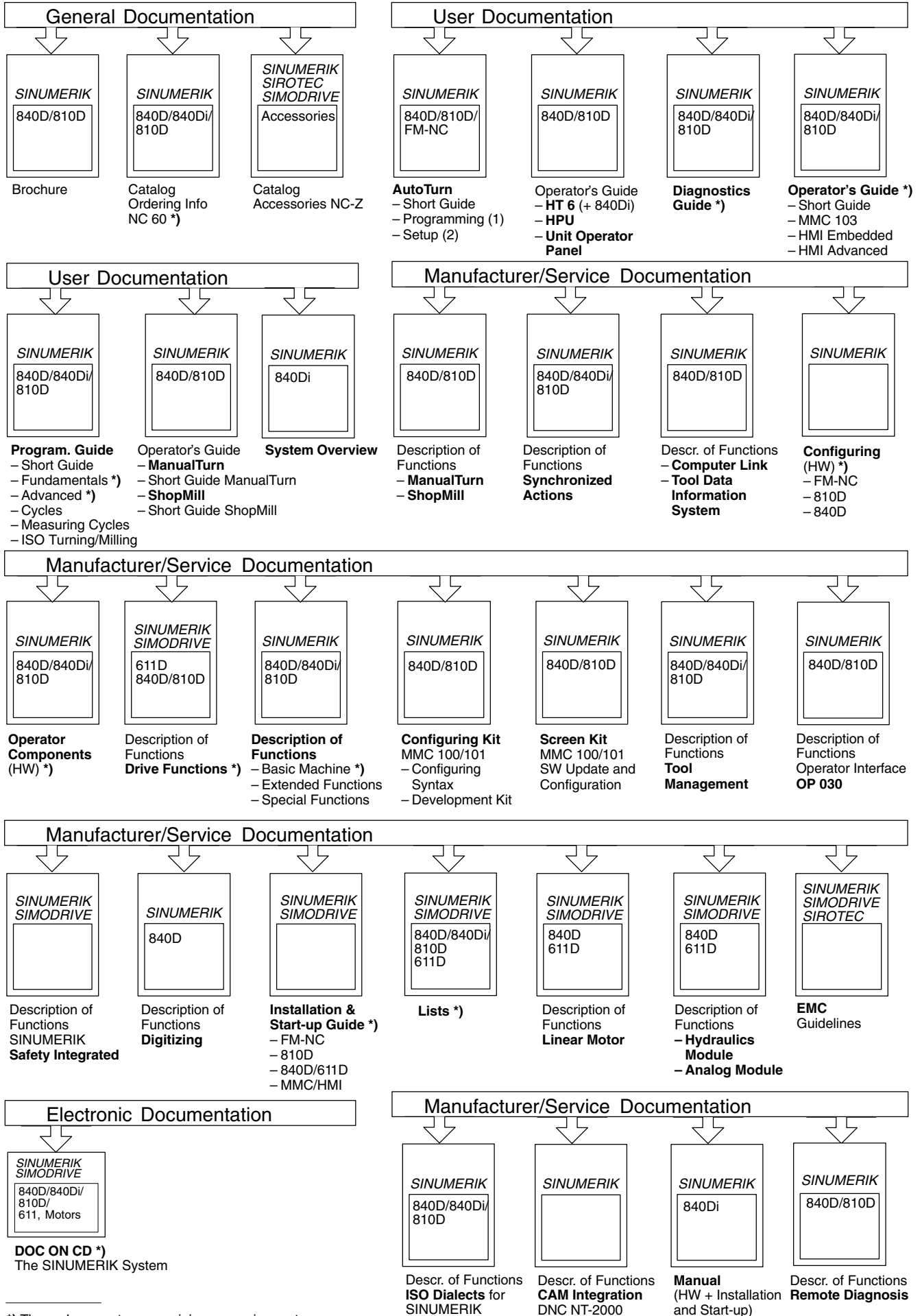
D-91050 Erlangen, Germany

(Tel. ++49 – 180 / 5050 – 222 [Hotline]
 Fax: ++49 – 9131 / 98 – 2176
 email: motioncontrol@erlf.siemens.de)

Sender		Suggestions
Name		Corrections
Address of your company/Department		For printed matter:
Street		SINUMERIK 840D/810D Configuring the OP 030 Operator Interface
Post code	Town:	Manufacturer/Service Documentation
Telephone:	/	Description of functions
Fax:	/	Order No.: 6FC5 297-6AC40-0BP0 Edition: 09.01
		If you notice any printing errors when reading this documentation, please use the preprinted form to advise us of these. We would also be grateful for ideas and suggestions on how we can improve this documentation.

Suggestions and/or corrections

Overview of SINUMERIK 840D/840Di/810D Documentation (08.01)



*) These documents are a minimum requirement