topline

Manual & Practice | PCS maxi

PCS 9000 / PCS 9100

maxi Operating Console

Multitasking window system ■
Intelligent soft-key action ■
Logging • Statistics • Reporting ■

The operating console PCS*topline* offers the highest degree on perfection unparalleled in design and function. With PCS*topline* everything is under control - from the PCS*mini* to the PCS*midi* up to the PCS*maxi*, with a superior operating culture and an unlimited setup freedom.

PCS, the first programmable operating console with a large selection of "ready-to-use" operating functions or operating tools which only need to be selected via instructions. Calmly, you can realize even odd operating requests in a minimum of time.

Today this way and tomorrow that way

One hardware standard for virtuel thousands of different operating situations. Without extensive wiring and dozens of I/O points.

PCS for operating. What else?

Print 03/06.94

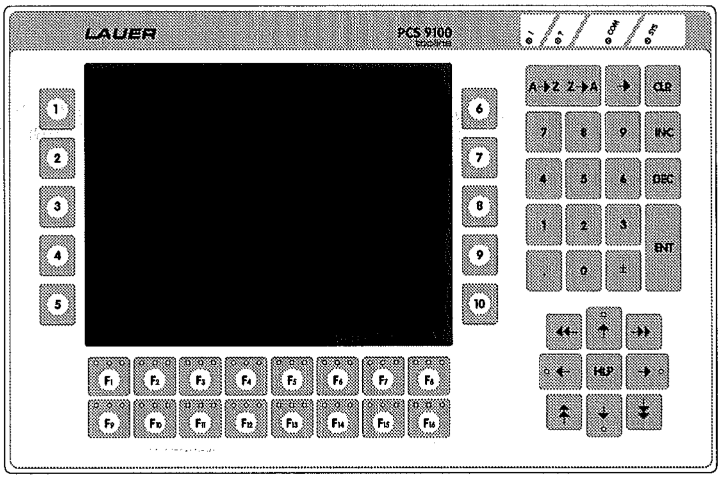
Contents

Page

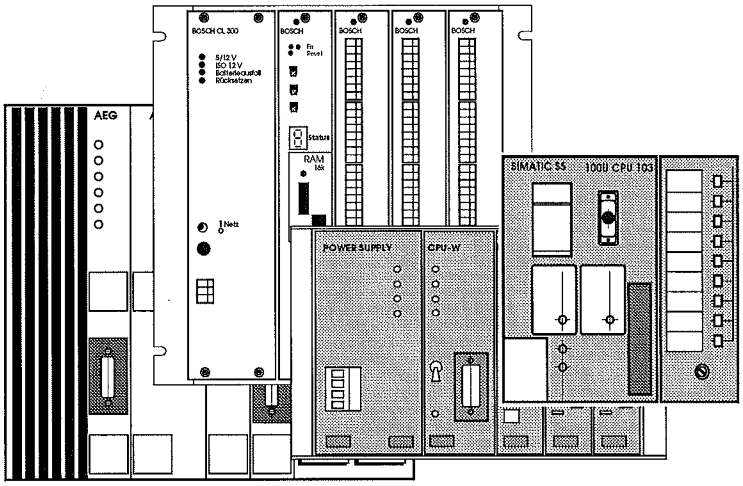
1 Complete Automatization 3
2 PCS 9000 / PCS 9100 4
3 The connections and modules 6
4 Specifications and dimensions 8
5 *Multitasking Window System* 9
6 Control and cursor keys for preset value input 10
7 Language selection and character set 11
8 Graphics for images and variables 13
9 Programming and communication 14
10 The easy programming principle 15
11 Data words and data bit allocation 16
12 Functions and tools 28
13 The different information media 29
14 Page descriptions ....
❶ Idle page 30
❷ Soft-key labeling 31
Soft-key actions 32
❸ Status page 34
❹ Headline page 35
❺ Message page 35
❻ Operating page/menu 37
❼ Help page 38
❽ Operator printer page 39
Printer interface 41
15 Static elements 42
❶ Static text 43
❷ Static bitmap 43
❸ Pixel 43
❹ Lines 45
❺ Rectangles 46
❻ Filled rectangles 46
❼ Circles 46
16 Variables used as dynamic elements 47
17 Overview internal and external variables 48
18 Variable formats 50
19 Date and time 56
20 The project planning software PCS 9092 57
21 Preset value entry using a clear menu structure 63
22 Quick start for setting up an operating project 65
23 Positionable elements 66
24 Data record example for an operating project 68
25 Example for planning a project in practice 69
26 Automux 97
27 Quality and Support 98
28 Index 99
29 Infos 102

Complete Automatization

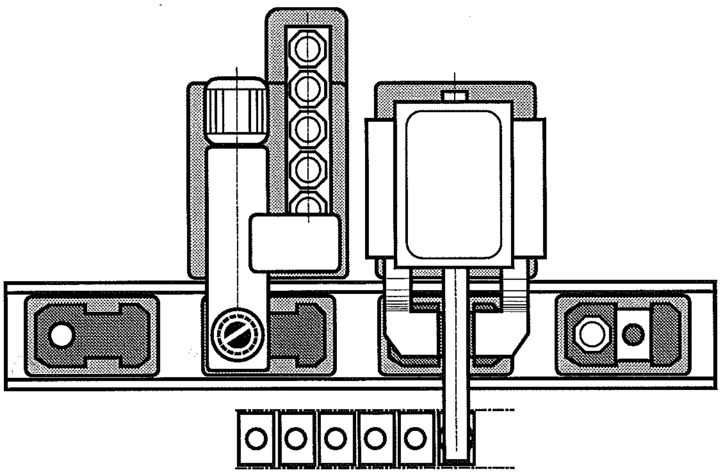
Operating & Monitoring



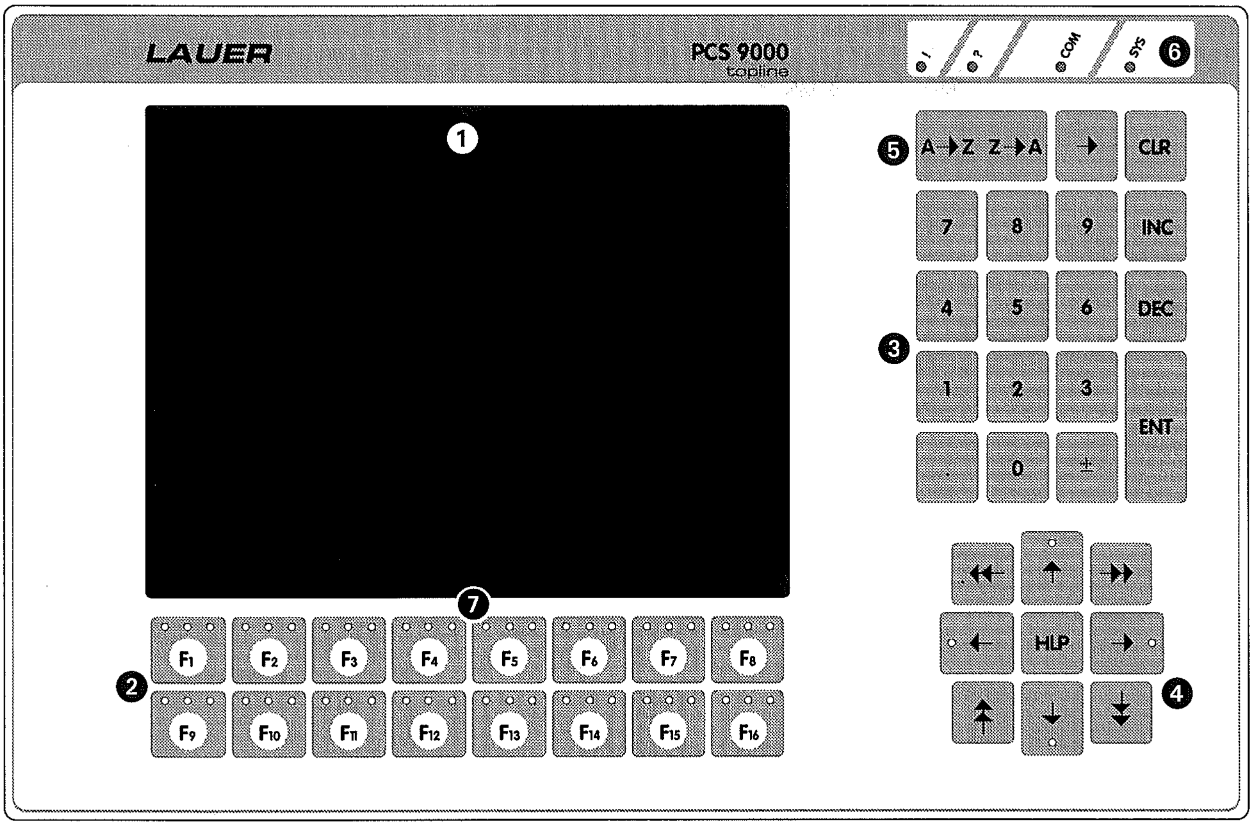
Control



Machine



2 The *maxi* operating console PCS 9000



❶ active LCD display with 640 x 480 pixels, full graphics capabilities ❷ function keys F1...F16 also for soft-key actions with 3 indicator LEDs ❸ numeric keypad for preset values ❹ 9 control keys for menu selections and preset value input, cursor keys with LEDs ❺ alpha-numeric keyboard for preset values ❻ important information on the PCS status ❼ function key labeling via display

The modular operating console PCS 9000 with an active LCD display offers the highest setup freedom.

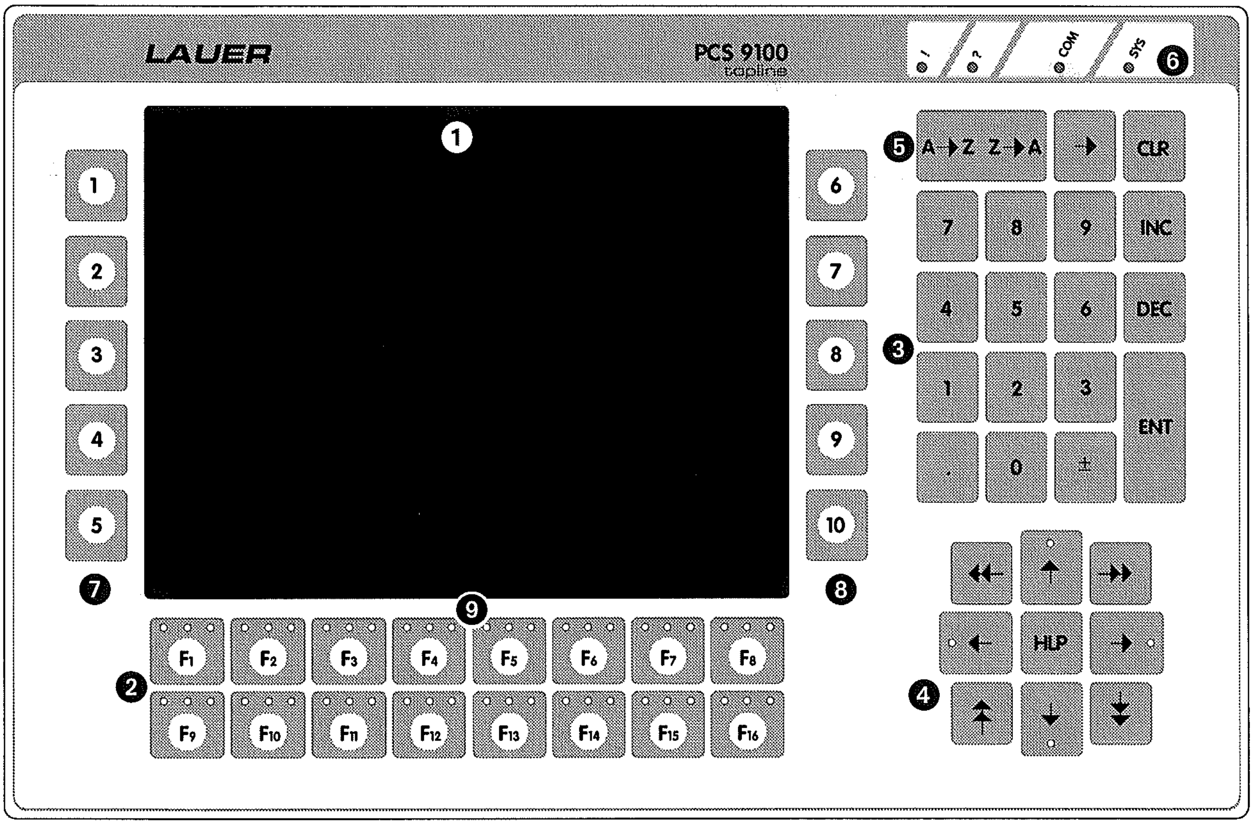
The display allows free 2D text and graphic representations for information and variable diplays. The **Multitasking Window System** (MWS) of the PCS*maxi* manages and updates four free-configurable windows for soft-keys, idle pages, messages, states, menus, recipes, and help information in parallel.

Managed by in an internal bus four modules with different functions can be plugged-in on the rear side. This is a precondition for the extremely flexible use of the PCS*maxi*.

PCS Status (❻)

! !	Message/menu disabled Prog. contr. disables messages/ menus
?	Operator inquiry, an input is expected
COM COM	No communication yet Coomunication is interrupted
SYS SYS	PCS without program or empty cassette is plugged-in Program transfer (!, ?, COM)

COM, ! = LED static, COM, ! = LED flashing

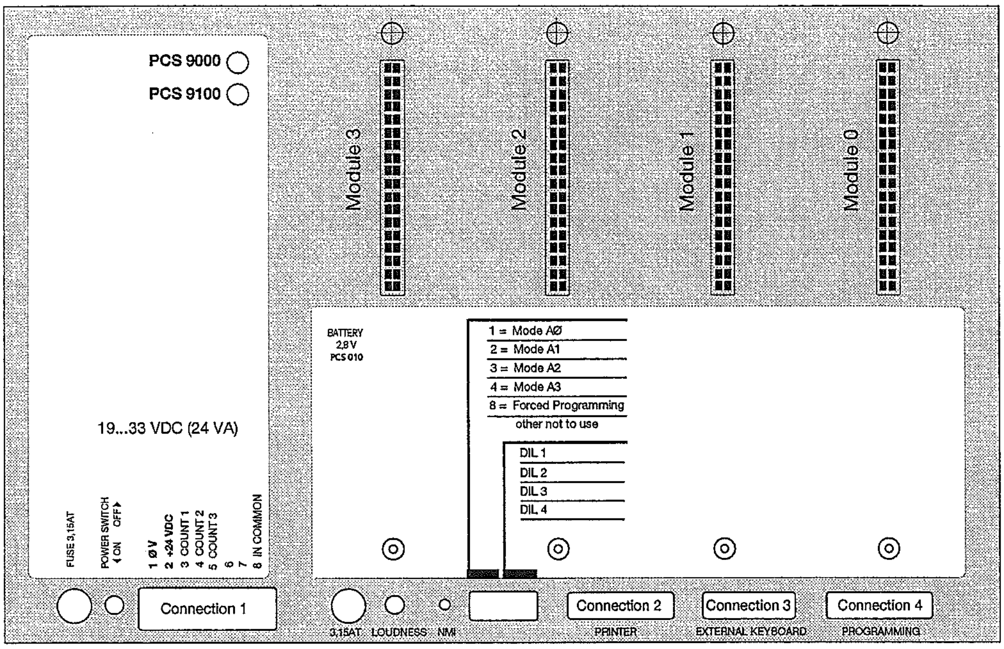


❶ active LCD display with 640 x 480 pixels, full graphics capabilities ❷ function keys F1...F16 also for soft-key actions with 3 indicator LEDs ❸ numeric keypad for preset values ❹ 9 control keys for menu selections and preset value input, cursor keys with LEDs ❺ alpha-numeric keyboard for preset values ❻ important information on the PCS status ❼ vertical soft-keys T1...T5 ❽ vertical soft-keys T6...T10 also for soft- key actions ❾ function key labeling via display

The modular operating console PCS 9100 with an active LCD display offers the highest setup freedom.

The display allows free 2D text and graphic representations for information and variable diplays. The **Multitasking Window System** (MWS) of the PCS*maxi* manages and updates four free-configurable windows for soft-keys, idle pages, messages, states, menus, recipes, and help information in parallel.

Managed by in an internal bus four modules with different functions can be plugged-in on the rear side. This is a precondition for the extremely flexible use of the PCS*maxi*.



Port 1 (8-pin terminal)

Pin	*Labeling*	*Function*	*Spezifications*
1,2	0V, +24V DC	Supply voltage:	19...33V DC
		Max. current consumption with cassette:	1.15 A
3, 4, 5	Count 1 ... 3	Count inputs 1, 2, 3:	+19...33V DC R_i = 7 kΩ f_z = 500 Hz @ 19V f_z = 1 kHz @ 24V
6,7		Floating contact (NO):	0,5A/50V resistive load
8	IN COMMON	0V for count inputs	

Port 2 (RS 232, 25-pin, female connector)

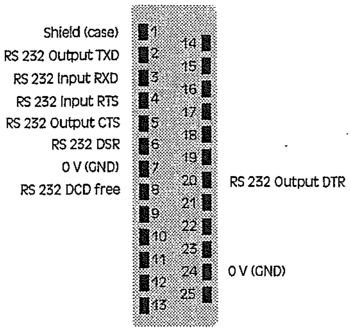
Printer or PC via the serial RS 232 interface. For the pin assignment see figure 1

Port 3 (RS 422, 15-pin, female connector)

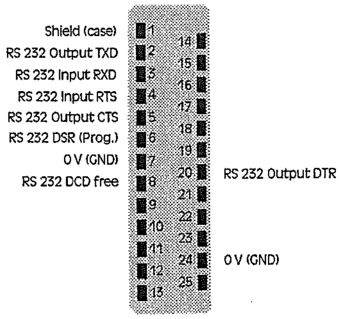
External keyboard. For the pin assignment see figure 3

Port 4 (RS 232, 25-pin, female connector)

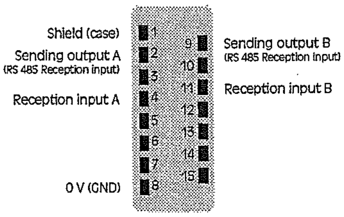
Programming of the PCS*maxi* via the serial RS 232 interface. For the pin assignment see figure 2



Pin assignment printer port and PCS 8010 (RS 232). Figure 1



Pin assignment programming port. Figure 2



Pin assignment external keyboard and PCS 8010 (RS 422, RS 485). Figure 3

The PCS*maxi* features 4 module slots on the rear side for various hardware configurations (memory expansion, logging, recipe, bar code reader, networking a.s.o). For the first time this results in a completely open hardware architecture realizing a multi-functionality for the PCS*maxi* and for operating in general. Many of the additonally required units and devices are thus eliminated which results in an evident cost reduction.

An interface module is mandatory. Other modules (please refer to the table) can be plugged-in and can be used in combinations. Use one of the programmable controller interface modules PCS 810 ... PCS 830.3 for the *Lauer* protocol.

		PCS 9000	PCS 9100	Slot 0	Slot 1	Slot 2	Slot 3
Interface module with RS 232, RS 422, RS 485, TTY [1]	PCS 8010	●	●			■	■
Interface module for Interbus S [1]	PCS 8020	●	●			■	■
Memory Pack 512 kB-PageROM	PCS 8100	●	●	■	■		
Memory Pack 2 MB-PageROM	PCS 8110	●	●	■	■		
Combi Pack 512 kB-PageROM for program 16 kB-EEPROM for recipe 64 kB-RAM for log	PCS 8120	●	●	■	■		
Interface module / Siemens	PCS 810	●	●				
Interface module / Siemens	PCS 810.3	●	●				
Interface module / Siemens (CP 523)	PCS 840	●	●				
Interface module / Klö Moe	PCS 820	●	●				
Interface module / Bosch	PCS 830	●	●				
Interface module / Bosch	PCS 830.3	●	●				
Automux for Siemens: S5 (L1- or AS 511 log)	PCS 809	●	●				
Programming cable PCS/PC [2]	PCS 733	●	●				
Adapter cable [3]	PCS nnn	●	●				
Project planning software [2]	PCS 9092	●	●				

[1] 1 interface module is mandatory for every prog. contr.   [2] 1 is only required   [3] matching cable

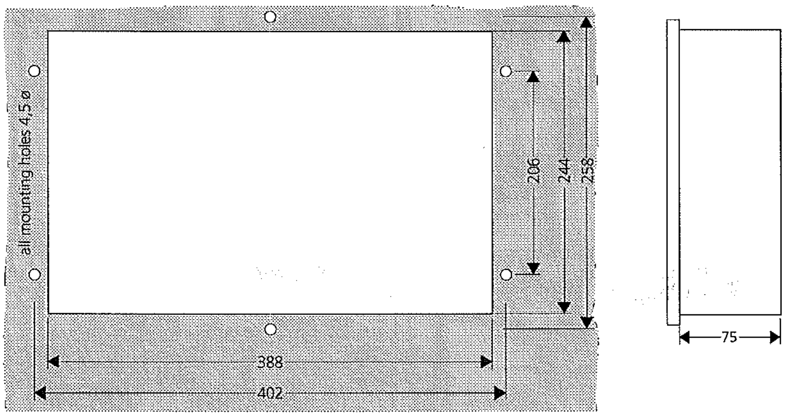
● can be used in combinations   ■ possible slots on the rear

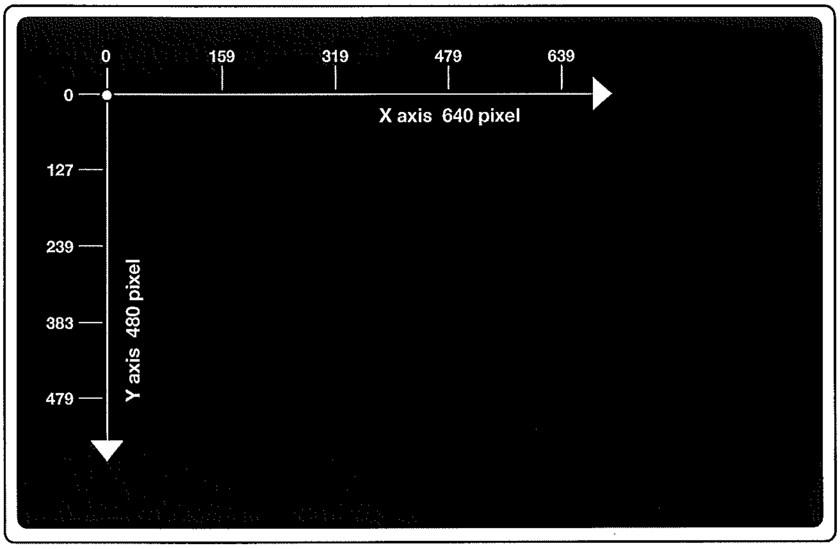
4 Specifications and dimensions

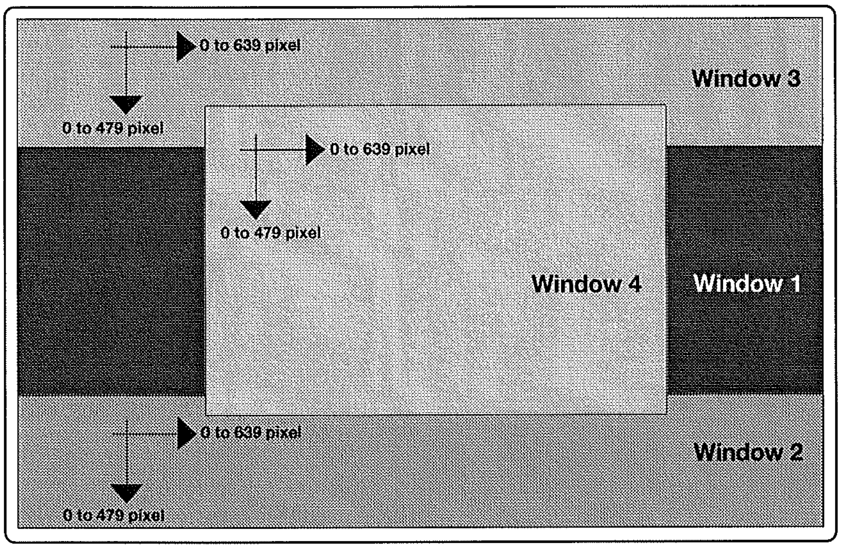
External dimensions:	410 mm x 266 m, depth without connector 75 mm
Weight:	approx. 3700 g; approx. 4700 g with 4 cassettes
Operating voltage:	+19...+33 V DC, reverse connection protected
Current consumption:	Iav without cassette @ 24VDC 650 mA Iav with 4 cassettes @ 24VDC 1.15 A Imax with 4 cassettes @ 32VDC (inrush current) 4 A
Internal memory:	Page ROM, 256 kByte approx. 80 KByte used for firmware
Data retension:	Page ROM, min. 100,000 write cycles

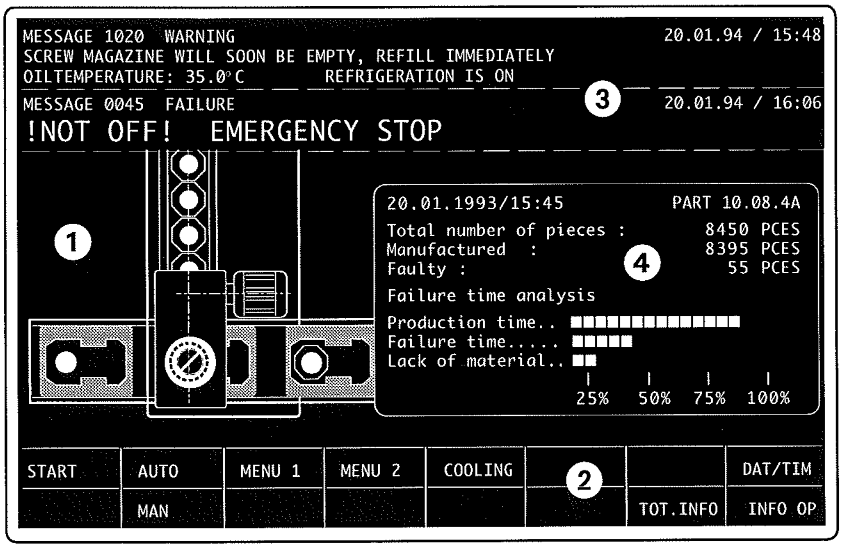
Count inputs:	logical 1: 16...32 V, logical 0: 0...5 V
	fz = 500 Hz @ 19 V, fz = 1 KHz @ 24 V
	Ri =7 kΩ

Floating output:	0.5 A/50 V resistive load (NO)
Printer port:	RS 232 (300...19200 Baud)
External keyboard:	RS 422 (130 kBaud)
Programming port:	RS 232 (38.4 kBaud)
Battery:	2.8 V, 1 Ah exchangable (life time 7 years @ 25°C)
Noise immunity:	Supply voltage IEC 801-4 4 KV RS 232 connection using a signal coupler IEC 801-4 2 kV discharge rear panel and front panel IEC 801-1 8 kV count inputs IEC 801-4 4 kV IEC 801-5 3 kV Exceeding these values will cause a controlled error message. A soldered-in pico fuse will burn-out bevore high voltages cause damages.
Protection class IEC 529:	rear panel IP 20, front side (in a built-in condition) IP 65
Humidity:	0...75%, min. 48 h exposure time
Vibration immunity:	2.5 g @ 50 and 75 Hz in all directions, min. 1 h.
Temperature:	storage -20...+60 °C operation 0...45 °C
Front side foil:	polyester
Keys	mechanical with tactile feed back, foil @ T1 ... T10
Display:	active LCD displaywith full graphics capabilities, 640 x 480 pixels
Fusing:	3.15 A, minature fuse slow-blow, 1 spare fuse









```
MESSAGE 1020  WARNING                                  20.01.94 / 15:48
SCREW MAGAZINE WILL SOON BE EMPTY, REFILL IMMEDIATELY
OILTEMPERATURE: 35.0°C        REFRIGERATION IS ON
MESSAGE 0045  FAILURE                                  20.01.94 / 16:06
!NOT OFF!  EMERGENCY STOP
```

```
20.01.1993/15:45                 PART 10.08.4A
Total number of pieces :           8450 PCES
Manufactured   :                   8395 PCES
Faulty  :                            55 PCES
Failure time analysis
Production time.. ■■■■■■■■■■■■■■■
Failure time..... ■■■■■
Lack of material.. ■■
                    |     |     |     |
                   25%   50%   75%   100%
```

The multitasking Window System of the PCS*maxi* manages and updates four free-configurable windows for soft-keys, idle pages, messages, menus, recipes, and help information in parallel.

In Window 1, one of the available 256 idle pages is displayed. With 640 x 480 pixels, the idle page occupies the complete area of the display. It is positioned in the background superimposed by other windows.

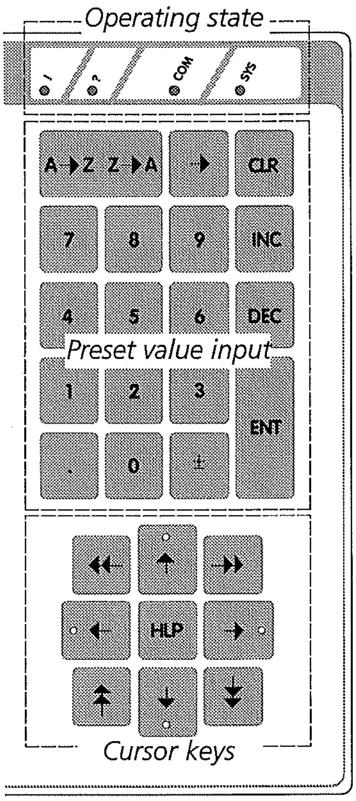
Window 2 can display 1 of the 256 soft-key assignment labelings. Generally, the soft-key labeling is displayed on the lower border of the display. The height is freely selectable in the range of 0 to 480. The width is fixed to 640 pixels.

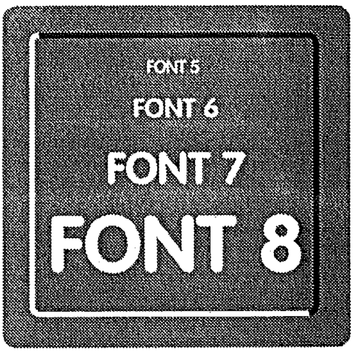
Window 3 is used for displaying the alternative status page or the message page. Generally, window 3 is located at the upper display border. The freely selectable height of the status page and the message page is in the range of 0 to 480 pixels. The status page and the message page have a fixed width of 640 pixels. The message page supersedes the status page.

Window 4 is used for the alternative display of 1 of the 255 possible menus, for the recipe management or for the display of additional information contained in a help page. The size and the position is selected by the user. A visible window 4 covers the windows 1 to 3.

VARIABLES FORMAT	KEYS	FUNCTION
BIT	INC	Sets bit in the word to a logical 1
GBIT	DEC	Sets bit in the word to a logical 0
	±	Toggles the bit: logical 0 / logical 1 / logical 0
	ARROWS*	Selects the next variable
STRING	INC*	Increments a STRING variable and sends the value to the prog. contr. @ CSTRING and GCSTRING
CSTRING	DEC*	Decrements a STRING variable and sends the value to the prog. contr. @ CSTRING and GCSTRING
GSTRING	ENTER	Sends the modified value to the prog. contr. (STRING/ CSTRING only)
GCSTRING	ARROWS*	Sends the value to the prog. contr. if it has been modified and selects the next variable (for CSTRING/GCSTRING same as bit)
	CLR	Recalls the previous string value (for STRING/GSTRING only)
BCD-1	INC*	Adds a value n within the limit values
BCD-2	DEC*	Subtracts a value n within the limit values
BCD0-1	CLR	Recalls the previous number value
BCD0-2	ENTER	Sends the modified value to the prog. contr.
	ARROWS*	Sends the value to the prog. contr. if it has been modified and selects the next variable
	DIGITS	Direct input of the number value
BIN-1/BIN0-1		Same as BCD
BIN-A/BIN0-A	POINT	Changes to the decimal digits if a point has been defined (for BIN-1, BIN-2, BIN0-1, and BIN0-2 only)
BIN-2/BIN0-2		
BIN-B/BIN0-B		
VBIN-1/VBIN0-1		Same as BIN
VBIN-A/VBIN0-A	±	Toggles the sign
VBIN-2/VBIN0-2		
VBIN-B/VBIN0-B		
WORD	0	Bit is set to "0"
	1	Bit is set to "1"
	INC*	Cursor is positioned to the right
	DEC*	Cursor is positioned to the left
	ENTER	Sends modified value to the prog. contr.
	CLR	Recalls the previous bit pattern
	ARROWS*	refer to STRING
ASCII	A -> Z	Increments the ASCII codes
	Z -> A	Decrements the ASCII codes
	——>	Next ASCII character, from left to right
	ENTER	Sends modified value to the prog. contr.
	CLR	Recalls the previous value
	ARROWS	refer to STRING

* = Auto-repeat





Up to 8 fonts

Up to 8 languages

With the elements Language selection, Character set and Font the PCS*maxi* features an extremly comfortable and a nearly unlimited text representation. Language, character set and font can be specified for each operating application individually.

Language selection

Operating applications can make use of a maximum of 8 languages. Using the global definition *LANGUAGEMAX* you define the number of languages for an application.

Character set

Each language features 2 character sets: small character set (Font 1) with 8 x 10 pixels
large character set (Font 2) with 16 x 20 pixels

Font 2 (large character set) is zoomable in 6 steps (Font 3 ... Font 8).

6 country specific character sets are available. Using the global definition the character sets are defined. If none has been defined the internal character set (PAGE 437) is used. Every character set selected with CHARSET occupies 15 kbyte per language of the PCS memory.

Language	*Character*	*Character set*		
		small	*large*	
from LANGUAGE 1				
	CHARSET (1),	437,	437	American
	CHARSET (1),	850,	850	Multi-language Latin 1
	CHARSET (1),	852,	852	Slavic Latin
	CHARSET (1),	860,	860	Portuguese
	CHARSET (1),	865,	865	Norwegian, Danish
	CHARSET (1),	866,	866	Russian
up to LANGUAGE 8				
	CHARSET (8),	437,	437	American
	CHARSET (8),	850,	850	Multi-language Latin 1
	CHARSET (8),	852,	852	Slavic Latin 2
	CHARSET (8),	860,	860	Portuguese
	CHARSET (8),	865,	865	Norwegian, Danish
	CHARSET (8),	866,	866	Russian
		Font 1	*Font 2*	*(font 2 is zoomable up to font 8)*

Example

An operating application is targeted for 3 languages. Each language requires a different character set. The following statements are needed in the global definition:

LANGUAGEMAX,(3)
CHARSET (1),437,437 German
CHARSET (2),850,850 English
CHARSET (3),437,437 Italian

Country	Character sets	
Australia	437	850
Austria	437	850
Belgium	437	850
Brazil	437	850
Canada (engl.)	437	850
Canada (french)	863	850
Denmark	865	850
Finland	437	850
France	437	850
Germany	437	850
Great Britain	437	850
Hungary	852	850
Israel	862	850
Italy	437	850
Japan	850	
Korea	437	
Latin America	437	850
Netherlands	437	850
Norway	865	850
Poland	852	850
Portugal	860	850
Russia	866	850
Sweden	437	850
Switzerland	437	850
Spain	437	850
Tschechoslowakia	852	850
Turkey	850	
USA	437	850
Yugoslavia	852	850

Using the internal string variable [LANGUAGE]*, a simple menu for* ***language selection*** *can be implemented.*

Example

The following is valid for global definitions

```
LANGUAGEMAX,(3)
CHARSET (1),437,437    German
CHARSET (2),850,850    English
CHARSET (3),437,437    Italian
```

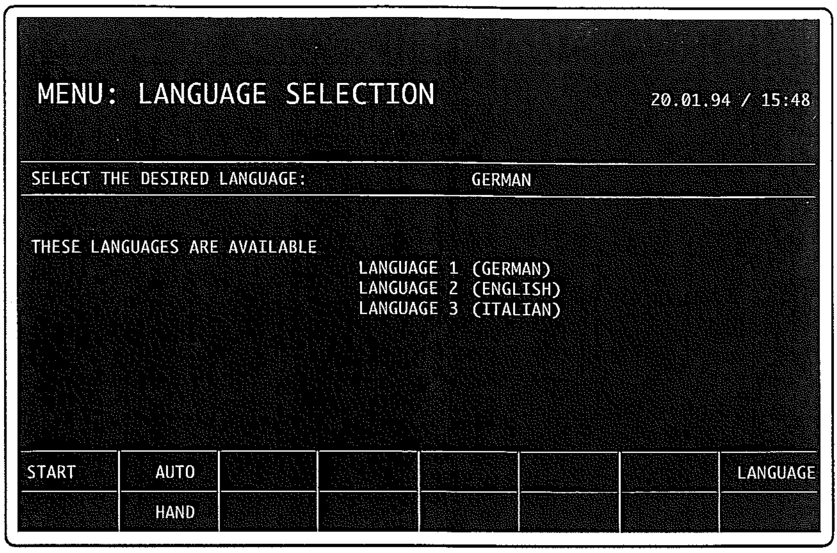
The language selection menu could have the apperance of figure 1. The internal STRING variable [LANGUAGE] is parameterized for a selection of 3 languages as follows:

```
INTVAR, [LANGUAGE], 3
STATE_TEXT, 0
LANGUAGE (1), "DEUTSCH"
LANGUAGE (2), "GERMAN"
LANGUAGE (3), "TEDESCO"
STATE_TEXT, 1
LANGUAGE (1), "ENGLISCH"
LANGUAGE (2), "ENGLISH"
LANGUAGE (3), "INGLESE"
STATE_TEXT, 2
LANGUAGE (1), "ITALIENISCH"
LANGUAGE (2), "ITALIAN"
LANGUAGE (3), "ITALIANO"
```

With F8 the user selects the Language selection menu. The PCSmaxi responds with figure 1. The INC key or DEC key is used to select the language "Italian". Figure 2. Automatically, the PCSmaxi shows all texts in italian, if selection is confirmed with RETURN. Reselecting the Language menu will show all texts in italian. Figure 3.

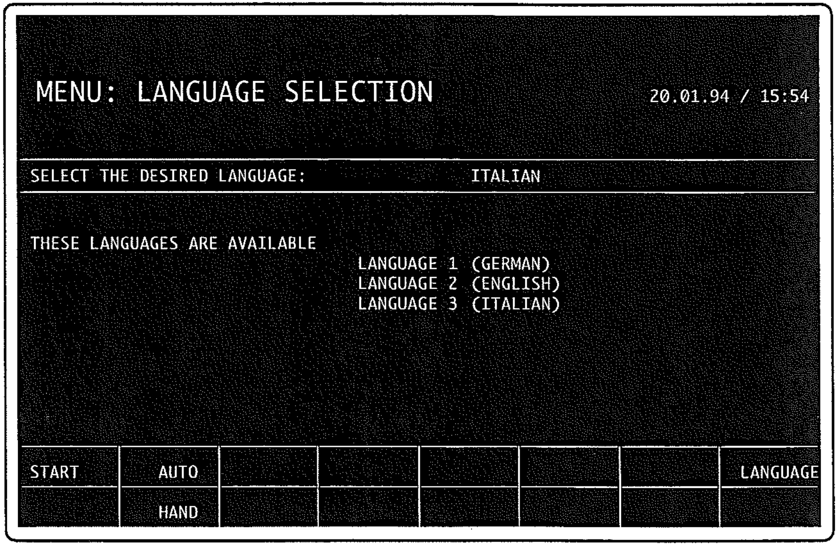
Since the font size can be freely selected for the PCSmaxi, it must be specified every time. Example:

```
OP_PAGE,1 , (624,300)
INS_ST_TEXT,24, 53,FONT(2)
LANGUAGE (1),"MENÜ: AUSWAHL DER SPRACHE"
LANGUAGE (2),"MENU: LANGUAGE SELECTION"
LANGUAGE (3),"MENU: SCELTA DELLA LINGUA"
INS_ST_TEXT,24, 128,FONT(1)
LANGUAGE (1),"WÄHLEN SIE DIE RICHTIGE..."
LANGUAGE (2),"SELECT THE RIGHT..."
LANGUAGE (3),"SCEGLIERE LA LINGUA..."
INS_INTVAR, [LANGUAGE], (314, 128), FONT(1)
OP_PAGE_END
```



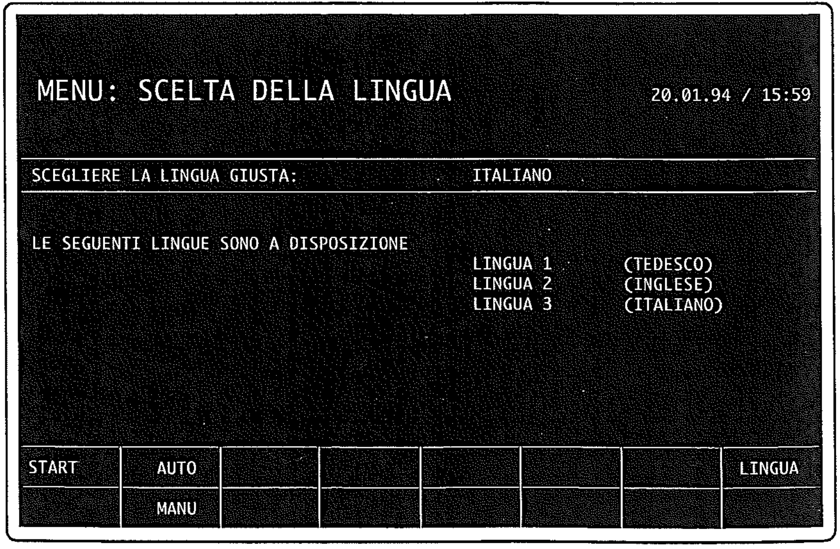
```
MENU: LANGUAGE SELECTION                          20.01.94 / 15:48

SELECT THE DESIRED LANGUAGE:                  GERMAN

THESE LANGUAGES ARE AVAILABLE
                                 LANGUAGE 1 (GERMAN)
                                 LANGUAGE 2 (ENGLISH)
                                 LANGUAGE 3 (ITALIAN)
```

FIGURE 1



```
MENU: LANGUAGE SELECTION                          20.01.94 / 15:54

SELECT THE DESIRED LANGUAGE:                  ITALIAN

THESE LANGUAGES ARE AVAILABLE
                                 LANGUAGE 1 (GERMAN)
                                 LANGUAGE 2 (ENGLISH)
                                 LANGUAGE 3 (ITALIAN)
```

FIGURE 2



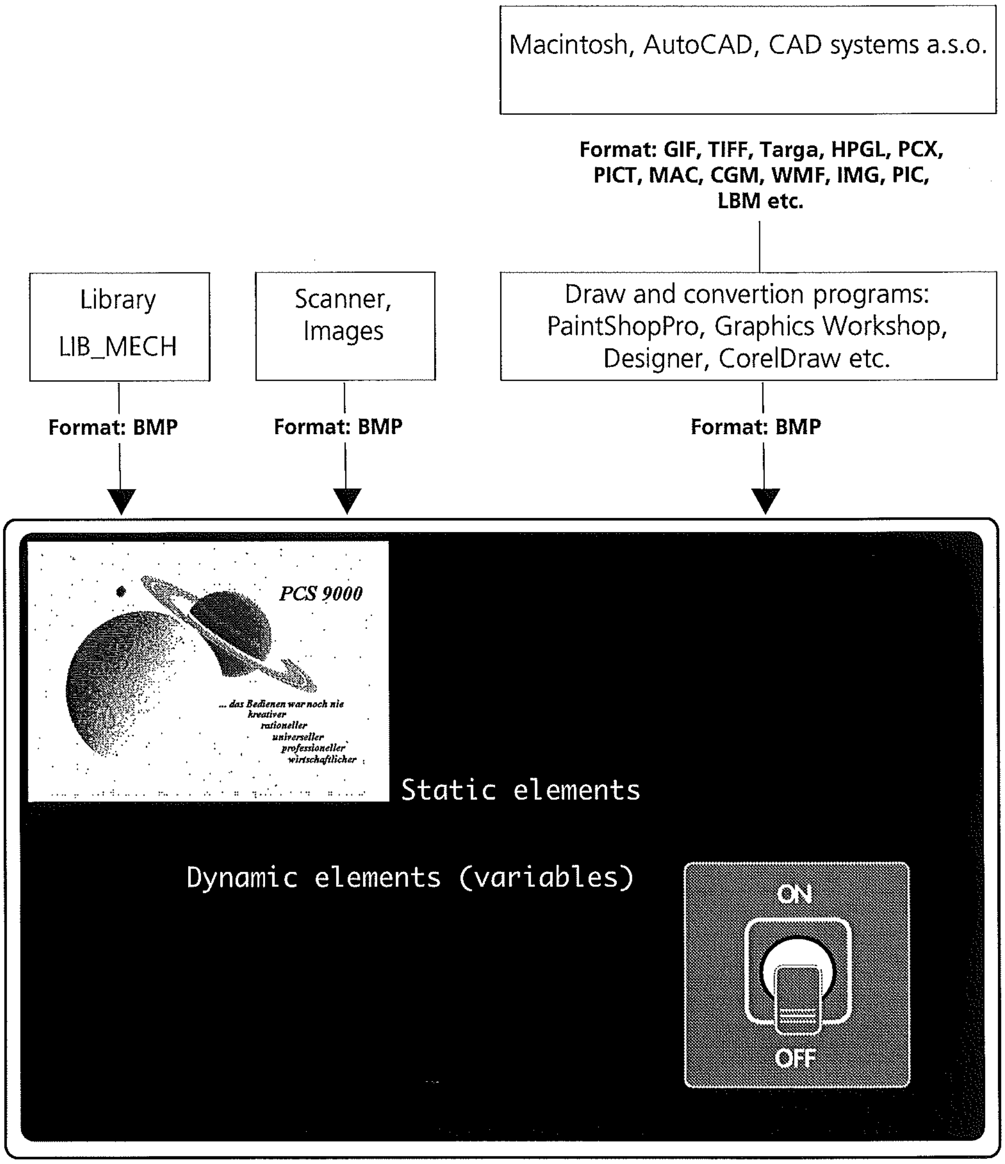
```
MENU: SCELTA DELLA LINGUA                         20.01.94 / 15:59

SCEGLIERE LA LINGUA GIUSTA:                   ITALIANO

LE SEGUENTI LINGUE SONO A DISPOSIZIONE
                                      LINGUA 1    (TEDESCO)
                                      LINGUA 2    (INGLESE)
                                      LINGUA 3    (ITALIANO)
```

FIGURE 3

Graphics for images and variables 8

Bitmaps originating from CAD systems, scanners, or visualizing programs and containing diagrams, sequential function charts and images can be included and displayed with the PCS*maxi*. Using graphical variables will animate still images. Process events and manufacturing steps are displayed synchronous to machine states.

Graphics display



Macintosh, AutoCAD, CAD systems a.s.o.

Format: GIF, TIFF, Targa, HPGL, PCX, PICT, MAC, CGM, WMF, IMG, PIC, LBM etc.

Library LIB_MECH

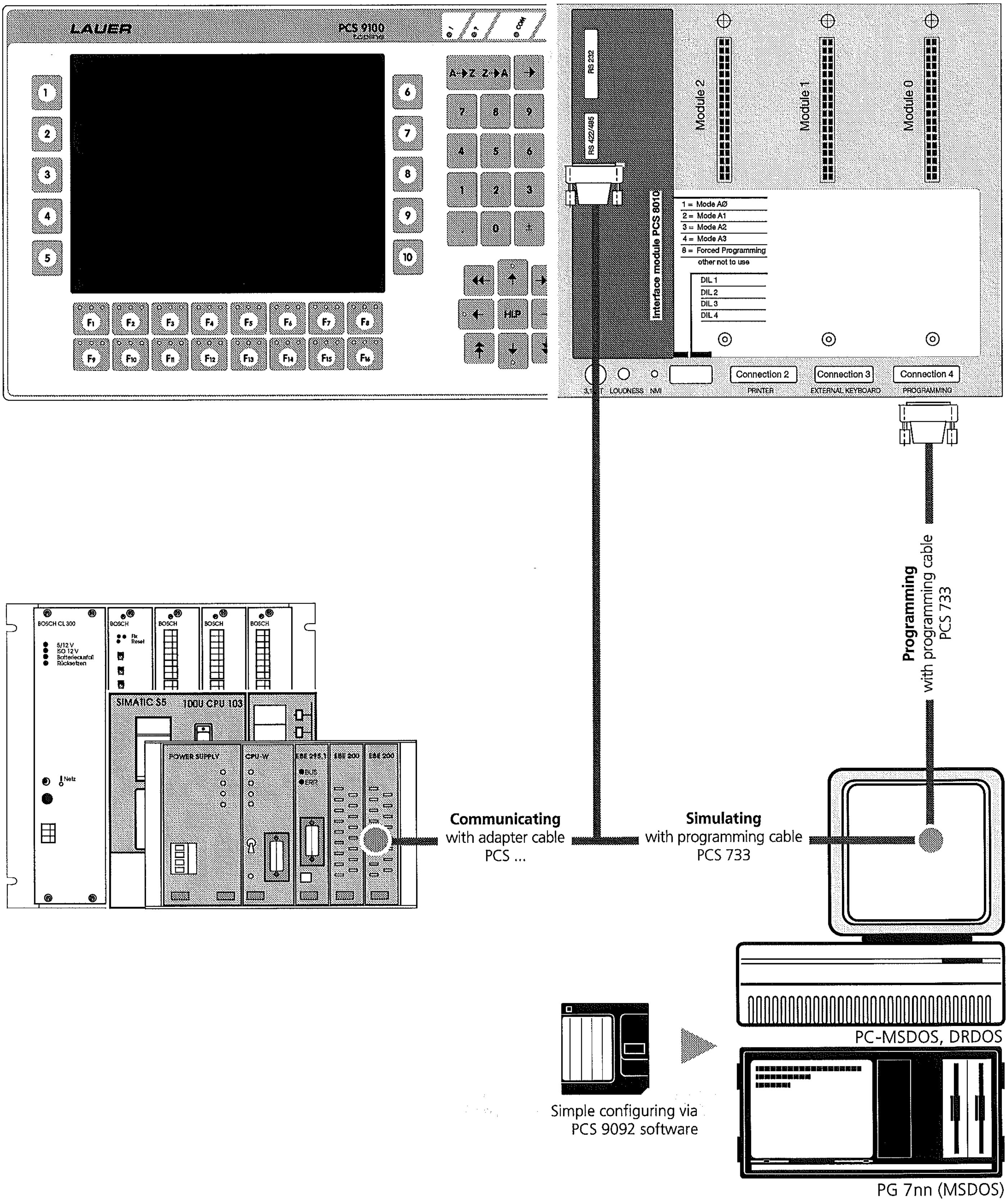
Scanner, Images

Draw and convertion programs: PaintShopPro, Graphics Workshop, Designer, CorelDraw etc.

Format: BMP

Format: BMP

Format: BMP



Communicating
with adapter cable
PCS ...

Simulating
with programming cable
PCS 733

Programming
with programming cable
PCS 733

PC-MSDOS, DRDOS

PG 7nn (MSDOS)

Simple configuring via
PCS 9092 software

The easy programming principle 10

The electrial connection between an arbritrary programming controller and a PCS is made via a matching adapter cable PCS.

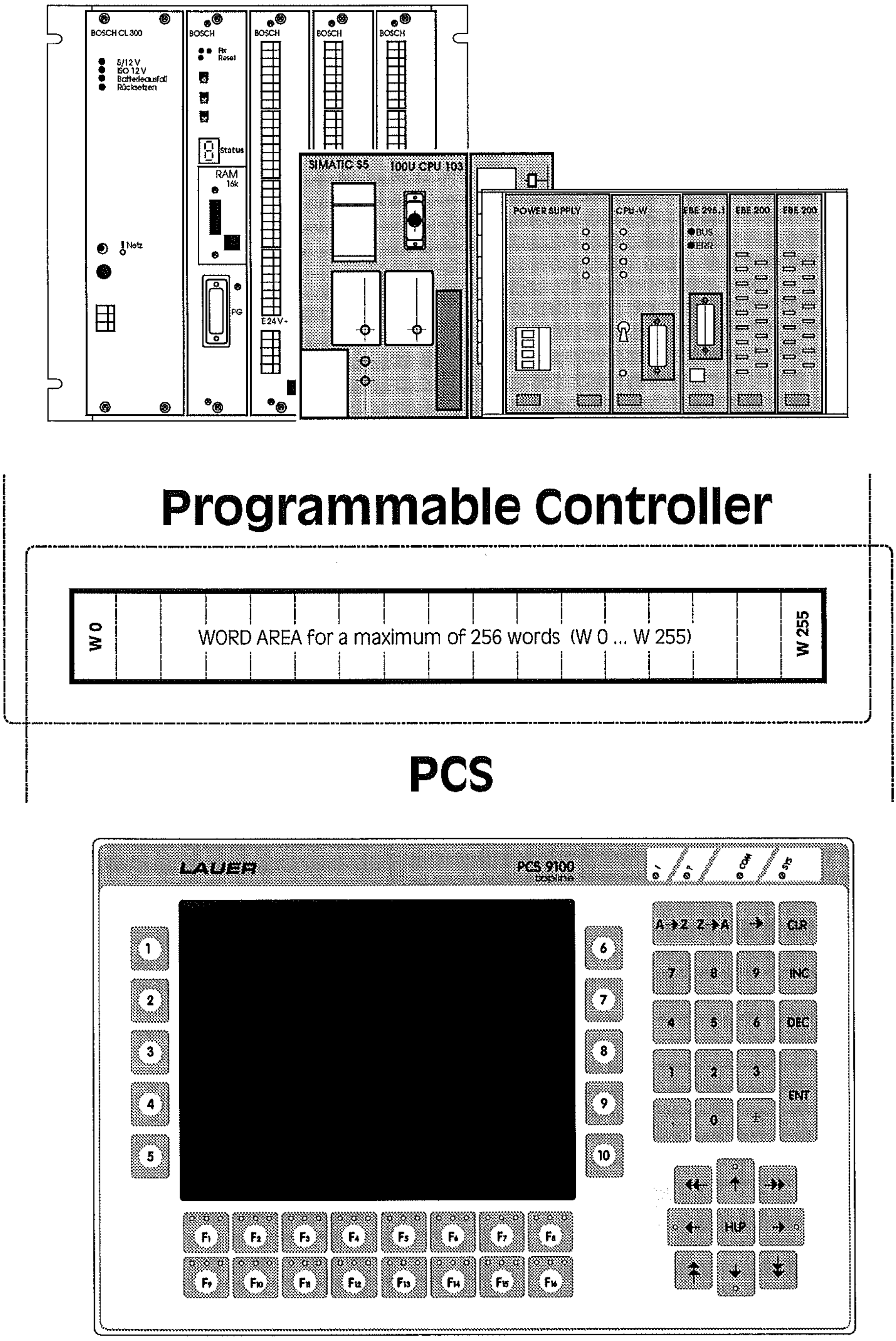
Data communication is based on an intelligible principle:

The PCS writes functions or preset values into predefined word areas of the programmable controller. The programmable controller then reads these words and interpretes them.

The programmable controller writes functions or actual values into predefined word areas. The PCS automatically reads and interpretes those word contents.

Depending on the type of the programmable controller, a maximum of 256 words with 16 bits each are available resulting in 4096 I/O points for PCS/ programmable controller communication.

The data words DW 00 ... DW 24 are reserved for the PCS*maxi* (please refer to page 16 ... 22). The data words 25 up to 255 are available for any operating application. The assignment of these data words is free.



11 Data words and data bit allocation

For the communication of the PCS*maxi* with an arbritrary programmable controller, data words (flags or simular) are used. A defined task or function is assigned to each data word DW:

Data word **Function** **PCS** **Prog. contr.**

Data direction

ERROR WORD

DW 03 Error word/synchronizing word - driver specific (please refer to the driver manual PCS 91.xxx)

KEY WORDS:

DW 04 Function keys F1...F16 ▶▶▶▶▶

DW 05 Vertical soft-keys T1...T10 (PCS 9100 only) ▶▶▶▶▶

DW 06 Control and cursor keys ▶▶▶▶▶

DW 07 Numeric keypad, DIL switch ▶▶▶▶▶

PCS STATUS WORDS:

DW 08 Priority state ▶▶▶▶▶

The currently highest priority is stored in bits 15...8.

Bit 15...8	*= 28*	*Recipe (highest priority)*
Bit 15...8	*= 24*	*Menu*
Bit 15...8	*= 20*	*Fault*
Bit 15...8	*= 16*	*Warning*
Bit 15...8	*= 12*	*Info*
Bit 15...8	*= 0*	*Idle page (lowest priority)*

The currently active priorities are stored in bits 7...0.

Bit 3	*Fault*
Bit 2	*Warning*
Bit 1	*Info*
Bit 0	*Menu*

DW 09 Page number with focus ▶▶▶▶▶

The page number with the highest priority is stored:

- *with help active the number of the help page (HLP_PAGE)*
- *with recipe active the number of the recipe page*
- *with a menu active the number of the operating page (OP_PAGE)*
- *with messages active the number of the message page (MSG_PAGE)*
- *otherwise the number of the idle page (REPORT_PAGE)*

Error word/synchronizing word driver specific

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 3 MB 6	Error word/synchronizing word driver specific								Error word/synchronizing word driver specific								MB 7

Function keys F1...F16

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 4 MB 8	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	MB 9

Vertical Soft-keys T1...T10 (only PCS 9100)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 5 MB 10	T1	T2	T3	T4	T5	X	X	X	T6	T7	T8	T9	T10	X	X	X	MB 11

Control and cursor keys

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 6 MB 12	↑	↓	←	→	↑↑	↓↓	← ←	→ →	A->Z	Z->A	->	CLR	INC	DEC	ENT	HLP	MB 13

Numeric keypad, DIL switch

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 7 MB 14	0	1	2	3	4	5	6	7	8	9	•	±	DIL4	DIL3	DIL2	DIL1	MB 15

Priority status

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 8 MB 16	128	64	32	16	8	4	2	1	7	6	5	4	3 Fault	2 War-ning	1 Info	0 Menu	MB 17
	Active priority (focus) numeric								Active priority bit by bit								

Page number with focus

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 9 MB 18	X	X	X	X	X	X	512	256	128	64	32	16	8	4	2	1	MB 19

DW = Data word or
MB = Flag byte
depends on the programmable controller used and the handling module

Data word **Function** **PCS** **Prog. contr.**

Data direction

DW 10 Number of the menu node ▶▶▶▶▶
Basically, 0 is returned for menus with only 1 node

DW 11 Preset value word number, length of the preset value ▶▶▶▶▶
Bits 15...8 DW Number of the last modified preset value
Bit 7...0 Length in bytes of the last modified preset value
If the bits 7...0 = 0, it is a
BIT or GBIT variable

DW 12 Bit mask of the last modified BIT/GBIT variable ▶▶▶▶▶
DW 12 is only valid, if in DW 11 the bits 7...0 = 0

COMMAND WORDS:

DW 13 Transfer locking ◀◀◀◀◀
Locking the transfer reduces the response time of further transmitted values.

Bit 4 1 if no LEDs should be transmitted
Bit 1 1 if no messages should be transmitted
Bit 0 1 if no menus should be transmitted
Bit X 1 if no date information should be transmitted. The X position (0...15) is defined by the internal variable [CLK_DBIT] which can be specified optionally. Locking of the transfer is not possible, if the internal variable is not parameterized.
Bit X 1 if no time information should be transmitted. The X position (0...15) is defined by the internal variable [CLK_CBIT] which can be specified optionally. Locking of the transfer is not possible, if the internal variable is not parameterized.
Bit X 1 if no operating printer pages should be transmitted. The X position (0...15) is defined by the internal variable [OPPRT_SBIT] which can be specified optionally. Locking of the transfer is not possible, if the internal variable is not parameterized.

Menu node number

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 10 **MB 20**	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	**MB 21**

Preset value word length, Preset value length

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 11 **MB 22**	128	64	32	16	8	4	2	1	128	64	32	16	8	4	2	1	**MB 23**
		DW number of the last modified preset value								Length of the last modified preset value (byte)							

Bit mask of the last modified BIT/GBIT variable

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 12 **MB 24**	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	**MB 25**
		Bit mask HIGH byte (only for BIT/GBIT variables)								Bit mask LOW byte (only for BIT/GBIT variables)							

Transfer locking

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 13 **MB 26**	X	X	X	X	X	X	X	X	X	X	X	LED	X	X	Messages	Menu	**MB 27**

DW = Data word or
MB = Flag byte
depends on the programmable controller used and the handling module

Data word	Function		PCS Prog. contr. Data direction
DW 14	Display locking		◀◀◀◀◀
	Bit 5	*1 if no status page should be displayed*	
	Bit 4	*1 if no soft-key labeling should be displayed*	
	Bit 3	*1 if no faults should be displayed*	
	Bit 2	*1 if no warnings should be displayed*	
	Bit 1	*1 if no infos should be displayed*	
	Bit 0	*1 if no menus should be displayed*	
DW 15	Activating bits		◀◀◀◀◀
	Bit 10	*if set to 1, counter ZR2 is set to 0*	
	Bit 9	*if set to 1, counter ZR1 is set to 0*	
	Bit 8	*if set to 1, counter ZR0 is set to 0*	
	Bit 4	*set to 1 to activate the elasped-hour meter*	
	Bit 3	*set to 1 to enable the alarm output*	
	Bit 2	*if set to 1, positive pulses are counted at input ZR2* *if set to 0, counting is interrupted*	
	Bit 1	*if set to 1, positive pulses are counted at input ZR1* *if set to 0, counting is interrupted*	
	Bit 0	*if set to 1, positive pulses are counted at input ZR0* *if set to 0, counting is interrupted*	

Display locking

DW 14
MB 28

15	14	13	12	11	10	9	8
X	X	X	X	X	X	X	X

7	6	5	4	3	2	1	0
X	X	Status page	Softkey labeling	Fault	Warning	Info	Menu

MB 29

Activating bits

DW 15
MB 30

15	14	13	12	11	10	9	8
X	X	X	X	X	ZR2	ZR1	ZR0
					RESET		

7	6	5	4	3	2	1	0
X	X	X	Elapsed-hour meter	Alarm	ZR2	ZR1	ZR0
					RELEASE		

MB 31

DW = Data word or
MB = Flag byte
depends on the programmable controller used and the handling module

Data word	Function		PCS Prog. contr. Data direction
DW 16	Idle page number		◀◀◀◀◀
	Bit 7...0	*Number of the idle page to be displayed*	
DW 17	Access class, menu number		◀◀◀◀◀
	Bit 15...8	*Access class for locking variables in menus* *Access class = 0...255* *0 means variables are only displayed and cannot be edited (actual values).* *Only variables with a lower or the same access class (1...255) as entered here can be edited in the menu (preset value)*	
	Bit 7...0	*Number of the menu to be displayed* *Bit 7...0 = 0 if no menu is to be displayed*	
DW 18	Soft-key labeling number		◀◀◀◀◀
	Bit 7...0	*Number of the soft-key labeling to be displayed*	
DW 19	LED state left		◀◀◀◀◀
DW 20	LED state center		◀◀◀◀◀
DW 21	LED state right		◀◀◀◀◀
DW 22	LED flashing left		◀◀◀◀◀
DW 23	LED flashing center		◀◀◀◀◀
DW 24	LED flashing right		◀◀◀◀◀
DW 25...255	depending on the application		

Default page number

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 16 MB 32	X	X	X	X	X	X	X	X	128	64	32	16	8	4	2	1	MB 33

Access class, menu number

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 17 MB 34	128	64	32	16	8	4	2	1	128	64	32	16	8	4	2	1	MB 35
	Access class								Menu number								

Soft-key labeling number

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 18 MB 36	X	X	X	X	X	X	X	X	128	64	32	16	8	4	2	1	MB 37

LED status left

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 19 MB 38	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	MB 39

LED status center

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 20 MB 40	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	MB 41

LED status right

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 21 MB 42	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	MB 43

LED flashing left

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 22 MB 44	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	MB 45

LED flashing center

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 23 MB 46	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	MB 47

LED flashing right

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW 24 MB 48	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	MB 49

DW = Data word or
MB = Flag byte
depends on the programmable controller used and the handling module

Data word	Function	PCS Prog. contr. Data direction
MESSAGES:		
DW X_w	Message word 1	*depending on the clear behavior*
DW X_w+1	Message word 2	*depending on the clear behavior*
up to		
DW X_w+63	Message word 64	*depending on the clear behavior*

With the parameter MSG_DW_RANGE, X_{Start}, $X_{AnzahlDW}$ the range of the message words is defined

The message words must be in the range of DW 25 up to DW 255.

Example: MSG_DW_RANGE,25,10
=> DW 25 up to DW 34: 10 message words are allocated (10 message bits).

DATE:

DW X_w — Year ▶▶▶▶▶

Bit 15...12	*millennium*	*(1...2)*
Bit 11...8	*century*	*(0...9)*
Bit 7...4	*decade*	*(0...9)*
Bit 3...0	*year*	*(0...9)*

DW X_w+1 — Month, day ▶▶▶▶▶

Bit 15...12	*month (tenth)*	*(0...1)*
Bit 11...8	*month (unit)*	*(0...9)*
Bit 7...4	*day (tenth)*	*(0...3)*
Bit 3...0	*day (unit)*	*(0...9)*

By specifying the internal variable [CLK_DADR] optionally the two date words are allocated in the area of DW 25 up to DW 255. When no parameters are specified for the internal variable, the date cannot be read and it does not occupy any data words in the DB area (see also DW 13).

Example: INTVAR, [CLK_DADR], 30
allocates DW 30 and DW 31 for the date

DAY OF THE WEEK, TIME:

DW X_w — Day of the week, hour ▶▶▶▶▶

Bit 15...8	*day of the week*	*(0...6)*	*0 = monday*
Bit 7...0	*hour*	*(0...23)*	

DW X_w+1 — Minutes, seconds ▶▶▶▶▶

Bit 15...8	*minutes*	*(0...59)*
Bit 7...0	*seconds*	*(0...59)*

By specifying the internal variable [CLK_CADR] optionally the two words are allocated in the area of DW 25 up to DW 255. When no parameters are specified for the internal variable, the day of the week and the time cannot be read and do not occupy any data words in the DB area (see also DW 13).

Example: INTVAR, [CLK_CADR], 40
allocates DW 40 and DW 41 for the day of the week and the time

Messages

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW X_w MB X_B	M 15	M 14	M 13	M 12	M 11	M 10	M 9	M 8	M 7	M 6	M 5	M 4	M 3	M 2	M 1	M 0	MB X_B+1

to

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW X_w+63 MB X_B+126	M 1023	M 1022	M 1021	M 1020	M 1019	M 1018	M 1017	M 1016	M 1015	M 1014	M 1013	M 1012	M 1011	M 1010	M 1009	M 1008	MB X_B+127

Year

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW X_w MB X_B	MILLENNIUM				CENTURY				DECADE				YEAR				MB X_B+1

Month, day

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW X_w+1 MB X_B+2	MONTH (TENTH)				MONTH (UNIT)				DAY (TENTH)				DAY (UNIT)				MB X_B+3

Day of the week, hour

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW X_w MB X_B	DAY OF THE WEEK (0...6)								HOUR								MB X_B+1

Minutes, seconds

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW X_w+1 MB X_B+2	MINUTES								SECONDS								MB X_B+3

DW = Data word or
MB = Flag byte
depends on the programmable controller used and the handling module

Data word **Function** **PCS** **Prog. contr.**
Data direction

Operating printer pages

DW X_w Operating printer page ◀◀◀ ▶▶▶

Bit 7...0 Number of the operating printer page

The operating page number is entered (1..255). If the page has been printed completely, the PCS sets the bits 7...0 to a logical 0.

By specifying the internal variable [OPPRT_DM] optionally, the address is allocated in the area of DW 25 up to DW 255. When no parameters are specified for the internal variable, the operating printer pages are not printed and no data words are allocated in the DB area (see also DW 13).

Example: INTVAR, [OPPRT_DW], 50
allocates DW 50 for the request of operating printer pages

Operating printer pages

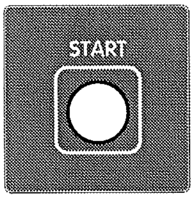
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
DW X_W **MB X_B**	X	X	X	X	X	X	X	X	128	64	32	16	8	4	2	1	**MB X_{B+1}**
									Number of the operating printer page								

DW = Data word or
MB = Flag byte
depends on the programmable controller used and the handling module

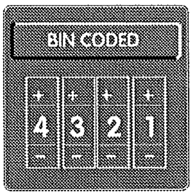
12 The functions and tools of the PCS*maxi*

PCS*maxi* is a homogeneous operating concept for different programmable controller systems. The operating consoles PCS 9000 and PCS 9100 feature a broad range of functions and tools for operating and monitoring:

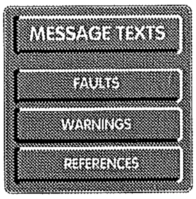
- *16 function keys with 3 message LEDs each green/red/green (OFF, ON, FLASHING, INVERSE FLASHING). More than 256 soft-key labelings can be freely specified (static and dynamic, text and graphics)*
- *An arbitrary number of switches with any annotation (text or graphics)*
- *An arbritrary number of selector switches with any annotation (text or graphics) and 256 switch positions each*
- *Key switch or code lock for assigning up to 256 different access rights*
- *Date and time, settable from the PCS or from the programmable controller for synchronization*
- *8 cam clocks with 8 cams each*
- *Digital BCD/BIN preset value entry via the numeric keypad or via the INC/DEC keys: An arbitrary number of preset values for 4 digit BCD, 5 digit BIN, 8 digit BCD or 10 digit BIN*
- *Easy ASCII preset value entry via the keys A->Z, Z->A and ->*
- *The representation and the modification of a word bit pattern (word variable) within the PCS is possible at any time*
- *Display binary actual values digitally, optionally up to 5 digits (0..65.535) or 10 digits (0...4.294.967.295)*
- *Automatic conversion of preset values and actual values from BCD/BIN into decimal and vice versa with sign, limit values and scaling*
- *1024 message pages with text and graphic variables with 3 message priorities and 2 clear modes*
- *256 pages with up to 64 variables each are available for idle pages*
- *Logging, statistics, machine report, output onto printer or PC*
- *255 menus with 64 variables each max. (16320)*
- *Display with 640 x 480 pixels and full graphics capabilities. Diagrams, function charts, images in the bitmap format of CAD systems, scanners or visualizing programs can be displayed*
- *Analog preset value input and analog actual value display via bargraphs 6 Bargraph formats: negative vertical bars, positive vertikale Bars, negative horizontal bars, positive horizontal bars, symmetrical vertical bars and symmetrical horizontal bars*
- *Up to 8 arbitrary languages with different character sets configurable, each language features 2 fonts (8 x 10 pixels and 16 x 20 pixels). The second font can be zoomed in 6 sizes. Thus font sizes of 8 x 10 up to 112 x 140 pixels can be displayed.*
- *Library of mechanical, electrical, and electronical BITMAP symbols conforming to DIN*
- *The* **Multitasking Window System** *(MWS) of the PCSmaxi manages and updates 4 freely configurable windows for soft-keys, idle pages, messages, status, menus, recipes, and help information*
- *Recipe manager for 255 recipes with 255 forms*
- *Soft-key action, 3 action groups with 255 actions each*



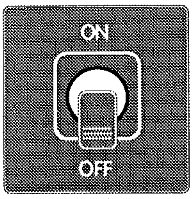
Keys



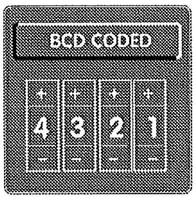
BIN preset value input



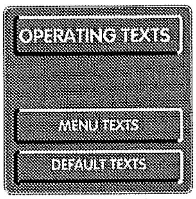
Message texts



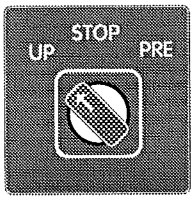
Switch



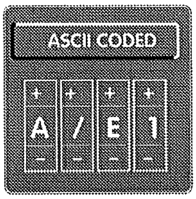
BCD preset value input



Operating and default texts

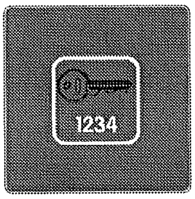


Selector switch

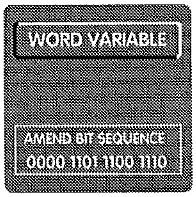


ASCII preset value input

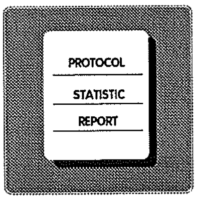
Help texts



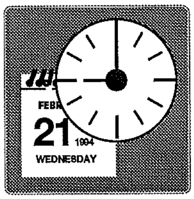
Code lock, key switch



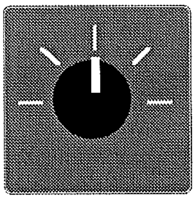
Change data word/flag

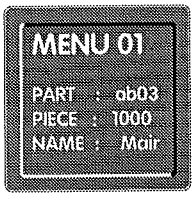


Protocol, statistic, report

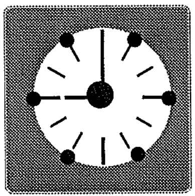


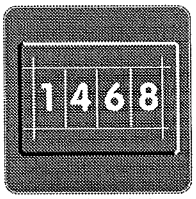
Date and time


Analog preset value input



Preset value input in a menu


Time clock

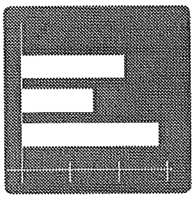


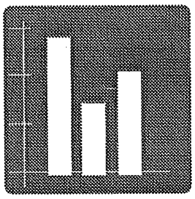
Digital actual value

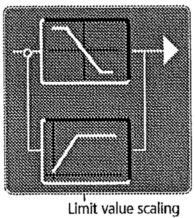
Graphics display

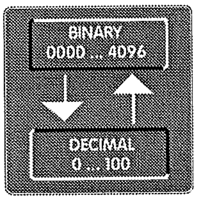
Up to 8 fonts

Up to 8 languages


Horizontal bargraph


Vertical bargraph


Limit value scaling



BIN/DEC conversion

Page description

Die PCS*maxi* offers 8 pages for various tasks and functions:

❶ Idle page (REPORT_PAGE)
❷ Soft-key labeling (SOFTKEY_ROW)
❸ Status page (STATUS_PAGE)
❹ Headline page (MSG_PAGE, TITLE)
❺ Message page (MSG_PAGE)
❻ Operating page/Menu (OP_PAGE/MENU)
❼ Help page (HLP_PAGE)
❽ Operator printer page (OP_PRNPAGE)

Static Elements

Bitmap	(INS_ST_BITMAP)
Text	(INS_ST_TEXT)
Circle	(INS_CIRCLE)
Point	(INS_PIXEL)
Line	(INS_LINE)
Rectangle	(INS_RECT)
Filled rectangle	(INS_FILLRECT)

Soft-key actions

With each soft-key labeling the function keys F1...F16 and T1...T10 can initiate specific actions - this applies to pressing the key as well as to releasing the key.

Each action can perform up to 8 operations (WRITE, AND, OR) of a constant with a DW (prog. contr.)

Dynamical elements

Internal variable

Depending on the individual firmware modules, the PCS creates different internal variables. Some examples:

[COM_ERRORS]
[COM_TIMEOUT]
[LANGUAGE]
[CLK_COUNT_H] a.s.o.

External variable

The prog. contr. and PCS read and write external variables together:

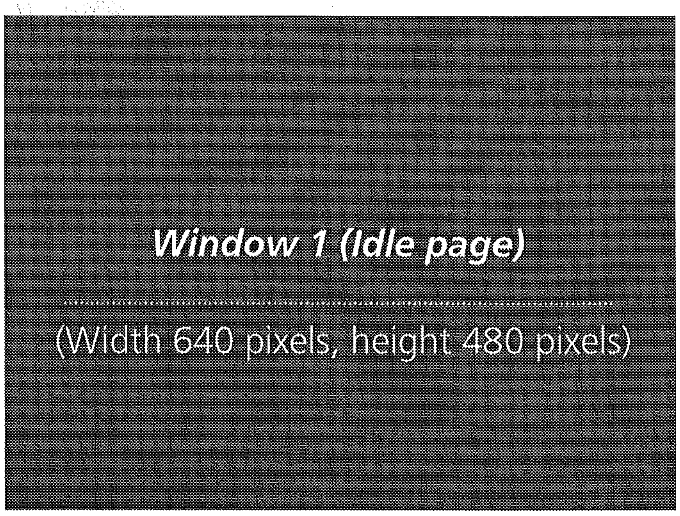
BIT, STRING, CSTRING, GBIT, GSTRING, GCSTRING, WORD, ASCII, BCD-1, BCD0-1, BCD-2, BCD0-2, BIN-1, BIN-A, BIN0-1, BIN0-A, BIN-2, BIN-B, BIN0-2, BIN0-B, VBIN-1, VBIN-A, VBIN0-1, VBIN0-A, VBIN-2, VBIN-B, VBIN0-2, VBIN0-B, PHBAR, NHBAR, PVBAR, NVBAR, SHSYMBAR, SVSYMBAR

❶ Idle page

256 idle pages are available to represent company logos and processes, to display information about machine states and to label the vertical soft-keys T1 ... T10 (PCS 9100).
The idle page is always located in window 1 and is covered by the other windows.

All dynamic and static elements can be positioned. Each idle page may contain up to 64 dynamic elements (variables) which are displayed but cannot be edited.

A DEFAULT idle page can be created to indicate idle pages not correctly loaded during startup. This DEFAULT idle page is displayed, if the programmable controller is loading an idle page which is not defined.



Example

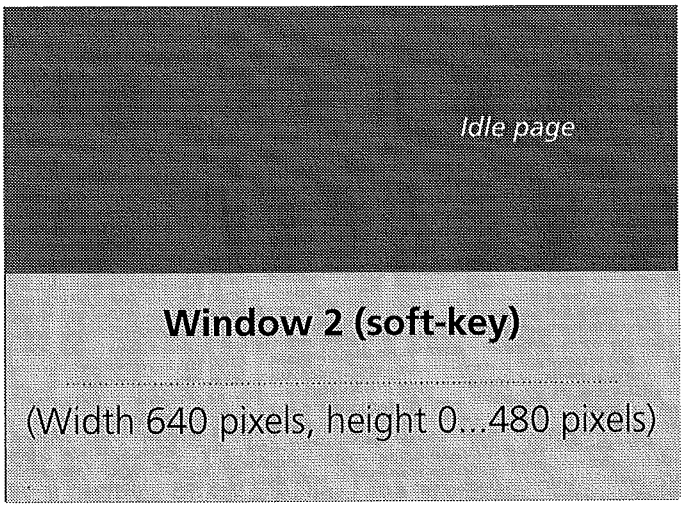
Page description of an idle page with static (text) and dynamic elements:

```
REPORT_PAGE, 0
INS_ST_TEXT, (30, 30), FONT (1)
LANGUAGE (1), "Text for idle page 0"
INS_INTVAR,[ CLK_DATE_ACT] ,(500,250), FONT (1)
REPORT_PAGE_END
```

Page description of a DEFAULT idle page:

```
REPORT_PAGE, DEFAULT
INS_ST_TEXT, (30, 30), FONT (1)
LANGUAGE (1), "Idle page not defined!"
REPORT_PAGE_END
```

❷ Soft-key labeling



256 soft-key labelings are available to label function keys F1...F16.

The soft-key labeling is always located in window 2 and covered by window 4.

All dynamic and static elements can be positioned. Each soft-key labeling may contain up to 64 dynamic elements (variables) which are displayed but cannot be edited.

A DEFAULT soft-key labeling can be created to indicate soft-key labelings not correctly loaded during startup. This DEFAULT soft-key labeling is displayed, if the programmable controller is loading a soft-key labeling which is not defined.

In the global definitions the soft-key labeling height (in pixels) is specified for each application.

Example: SKEYHIGH, (50)

Example

Page description of a soft-key labeling with bordering of the individual elements with static rectangles and lines. The soft-key labeling contains static (text) and dynamic elements:

```
SOFTKEY_ROW, 0
INS_RECT, (0, 49), (640, 49)
INS_LINE, (0, 25), (639, 25)
PATTERN (1)
INS_LINE, (80, 49), (80, 1)
PATTERN (1)
INS_LINE, (160, 49), (160, 1)
PATTERN (1)
INS_LINE, (240, 49), (240, 1)
PATTERN (1)
INS_LINE, (320, 49), (320, 1)
PATTERN (1)
INS_LINE, (400, 49), (400, 1)
PATTERN (1)
INS_LINE, (480, 49), (480, 1)
PATTERN (1)
INS_LINE, (560, 49), (560, 1)
PATTERN (1)
INS_ST_TEXT, (20, 18), FONT (1)
LANGUAGE (1), "Start"
SOFTKEY_ROW_END
```

Page description of a DEFAULT soft-key page:

```
SOFTKEY_ROW, DEFAULT
INS_ST_TEXT, (30, 30), FONT (1)
LANGUAGE (1), "A soft-key page has not been defined!"
SOFTKEY_ROW_END
```

Soft-key actions

Soft-key intelligence to reduce the programmable controller program and to increase response time.

Programmable controller programs for the management of idle pages, messages, menus, recipes and LEDs can be reduced by assigning arbritrary actions to the function keys F1 ... F16 and T1 ... T10 (PCS 9100).

Each function key may trigger different actions when pressed or released. For this purpose, PCS*maxi* offers the three action categories listed below.

❶ *Action 1.0 ... 255 (predefined)*
Category 1 only initiates actions in the PCS (internal)

❷ *Action 2.0 ... 255 (user defined)*
Category 2 only initiates actions in programmable controller (external)

❸ *Action 3.0 ... 255*
Category 3 initiates actions in the PCS and in the programmable controller

with 255 actions and up to 8 commands for each category

❶ Writing over (WRITE)

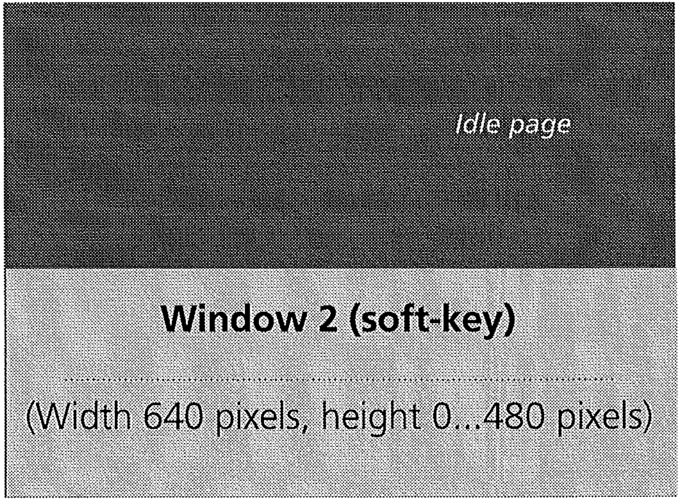
❷ Bit-by-bit AND, clear programmable controller bits

❸ Bit-by-bit OR, set programmable controller bits

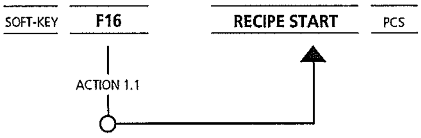
offering a nearly unlimited functional variety for selection.

Soft-key actions provide an enormous flexibility and can easily be understood.

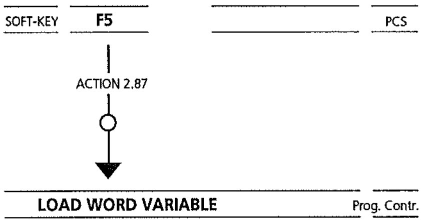
First of all, different actions are defined and then assigned to the function keys. As DEFAULT allocation, all actions are assigned to all 256 soft-key labelings, but it is possible to allocate special actions to each soft-key labeling.



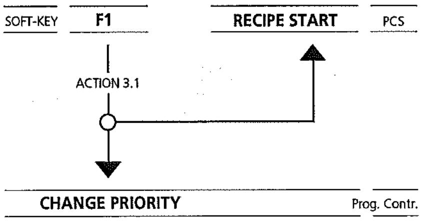
Action category 1



Action category 2



Action category 3



First of all, different actions are defined ...

***Example:** The action definition is separated from the soft-key page description.*
In data word 16, idle page 123 (decimal) and in data word 17, menu 1 with access class 1 (hexadecimal) is loaded. In data word 19, the left LED of function key F1 (binary) is activated and all other LEDs are deactivated. In the data words 20 and 21 the middle and right LED of function key F1 are deactivated while the status of the other LEDs is maintained:

```
ACTION, 45
Commands (WRITE, AND, OR)
 |       Data word
 |       |       Value in hex, dec, binary
 |       |       |
WRITE, 16,    123
WRITE, 17,    $0101
WRITE, 19,    b1000 0000 0000 0000
AND,   20,    b0111 1111 1111 1111
AND,   21,    b0111 1111 1111 1111
```

In data word 19, all left LEDs are deactivated, in data word 21 the right LED of function key F1 is activated while the other LEDs remain deactivated:

```
ACTION, 46
WRITE, 19,    0
OR,    21,    b1000 0000 0000 0000
```

In data word 16, idle page 200 is loaded.

```
ACTION, 47
WRITE, 16,    200
```

... then the actions are assigned to the function keys.

***Example:** Function key F1 is assigned action 45 when pressed and action 46 when released. Function key F2 is assigned action 47 when pressed.*

```
                   Function keys F1 ... F16, T1 ... T10
                   |         Group (category) 1 ... 3
                   |         |  Action at depressing
                   |         |  |  Group 1...3
                   |         |  |  |    Action at releasing
                   |         |  |  |    |
KEYDEFAULT, F1,          2 : 45, 2 : 46
KEYDEFAULT, F2,          2 : 47
```

In contrast to the DEFAULT assignments, another action sequence is assigned to the function keys F1 and F2 of soft-key labeling 125. F1 is assigned no action when released, while action 45 is maintained when the key is pressed.

```
SOFTKEY_ROW,125
Function keys F1 ... F16, T1 ... T10
 |       Group (category) 1 ... 3
 |       |  Action at depressing (0 = no action)
 |       |  |    Group 1...3
 |       |  |    |   Action at releasing
 |       |  |    |   |
F1,          ,  2 : 0
F2,      2 : 46, 2 : 45
SOFTKEY_ROW_END
```

Basic functioning of the soft-key actions

- After starting PCS*maxi* communication, soft-key labeling 0 and its function key assignment are assumed.
- While a function key is held down, no switching to another soft-key labeling occurs. This is done only after releasing the key and if a corresponding action has been defined.
- If the function keys are pressed faster than allowed by the programmable controller transfer time, the PCS*maxi* buffers up to 8 key operations.
- After defining actions which are to be executed when pressing and releasing a function key, the corresponding action is transferred to the programmable controller in two cycles.
- Key combinations (for example F1 plus F8) are admissible. The switching key F1, however, should not be assigned an action.
- The internal variables [STRT_KEYGRP] and [STRT_KEY-CODE] enable the action categories (1 ... 3) and the action (0 ... 255) of the last key operated to be displayed, if this was defined in a page description.

❸ Status page

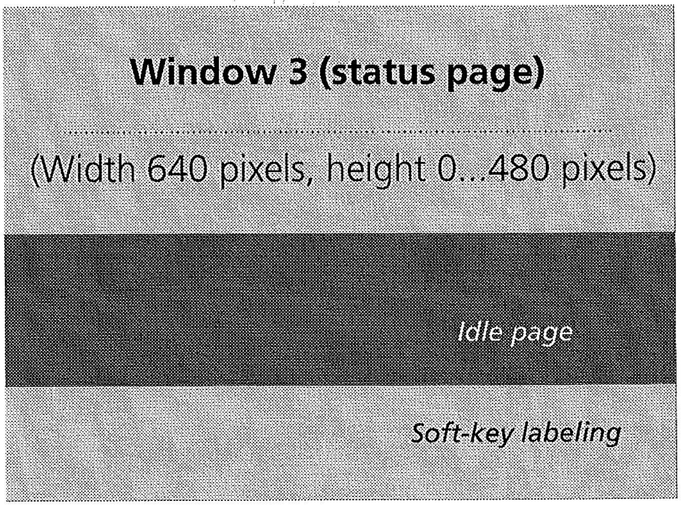
The status page contains information about the current machine and operation states (setting-up mode, automatic and manual mode), date, time, number of infos, warnings, faults, interface errors an others.

The status page is located in window 3 and covered by window 4.

All dynamic and static elements can be positioned as required. The status page may contain up to 64 dynamic elements (variables) which are displayed but cannot be edited.

The soft-key labeling height (in pixels) is specified for each application in the global definitions.

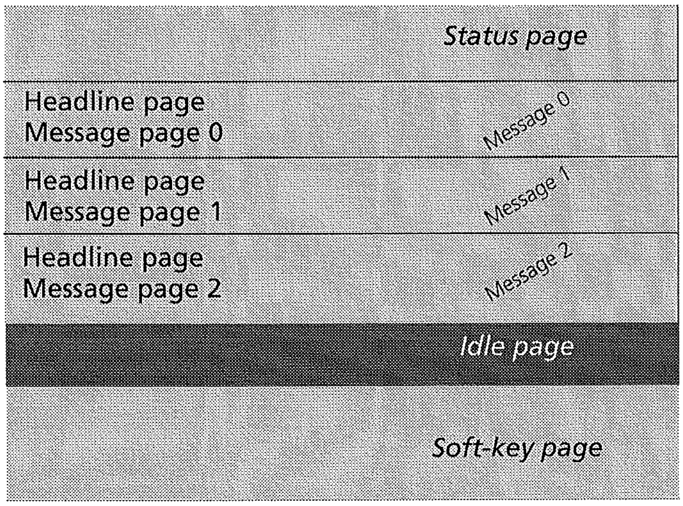
Example: STATUS_WINDOW, (50)



Example

Page description of a status page with static (text) and dynamic elements:

```
STATUS_PAGE
INS_ST_TEXT, (30, 30), FONT (1)
LANGUAGE (1), "Text of the status page"
INS_INTVAR,[ CLK_COUNT_HN] ,(40,250), FONT (1)
STATUS_PAGE_END
```



Example

Page description of a message page:

MSG_PAGE, 0, INFO, QUIT
STATUS_PAGE_END

Messages

Up to 1024 messages inform the operator about all relevant states. Each message is activated and deactivated by a programmable controller bit.

Messages are displayed below the status page in the message section of window 3. Window 3 may be covered by window 4.

❹ Headline page

All messages contain a headline page which is defined only once and used for all messages. The headline page may contain internal variables and static texts.

The internal variables [MSG_TCOME] for *MESSAGE COME* (when message bit is set to logical 1), [MSG_TQUIT] for *MESSAGE ACKNOWLEDGED* (CLR key pressed) and [MSG_TGONE] for *MESSAGE GONE* (when message bit is set to logical 0) are used in the heading page to indicate date and time of the events

Message received, Message acknolowedged,
Message gone

[MSG_TQUIT] and [MSG_TGONE] are only visible depending on the clear behavior.

❺ Message page

Each message page is assigned specific information which may contain all static and dynamic elements. Dynamic elements may contain up to 8 data words.

Each message (specified during message page description) is assigned a priority: info (**INFO**), warning (**WARNING**) or fault (**FAULT**). Faults have the highest and infos the lowest priority. In the PCS, messages of the highest priority are always displayed.

Example: ***3 infos and 2 faults are active (the corresponding bits are set in the programmable controller), but only the faults are displayed. The infos are visible, if the two faults have „gone".***

Each message page is also assigned a clear behavior:

NO_QUIT — The message may only be cleared by resetting the bit in the programmable controller.

QUIT — The message may be cleared by pressing the CLR key or by resetting the programmable controller bit.

The appearance of the messages is specified by the command

MSG_WINDOW, FONT (1), 15, 2

in the global definitions. FONT(1) determines the font and thus the height of the heading page. 15 specifies the height of the message page in pixels and 2 the number of messages displayed simultaneously.
The position of the bits in the transfer area are defined by

MSG_DW_RANGE, 25, 10

In the example, a total number of 160 message bits is displayed in the 10 data words from 25 to 34.

The number of data words determines the message refresh time and should cover the range of allocated message bits.

A DEFAULT message page may be defined for each priority (info, warning, fault) to indicate messages not correctly loaded during startup. The DEFAULT messages are displayed, if the programmable controller is loading non-defined messages.

If at least two priority-assigned messages are active and if window 4 is not displayed, the cursor keys enable paging between the different messages. If the cursor is positioned on a message which is assigned the clear behavior QUIT, the message is deleted by pressing the CLR key and the corresponding programmable controller bit is set to 0.

The HLP key is used to display specific help information for each message, if help pages have been parameterized.

Page description of a headline page: inside, the elements are specified.

```
MSG_PAGE, TITLE
INS_ST_TEXT, (12, 9)
LANGUAGE (1), "Message received:"
INS_INTVAR, [MSG_TCOME], (161, 9), FONT (1)
MSG_PAGE_END
```

Page description of a DEFAULT message page for the priority Fault.

```
MSG_PAGE, DEFAULT, FAULT, QUIT
INS_ST_TEXT, (155, 11), FONT (1)
LANGUAGE (1), "Message page"
INS_INTVAR, [MSG_TXTNR], (245, 11), FONT (1)
INS_ST_TEXT, (285, 11), FONT (1)
LANGUAGE (1), "not defined!"
MSG_PAGE_END
```

Page description of a message page 0.

```
MSG_PAGE, 0, INFO, QUIT
INS_ST_TEXT, (152, 12), FONT (1)
LANGUAGE (1), "Info, message 0, clear behaviour QUIT!"
MSG_PAGE_END
```

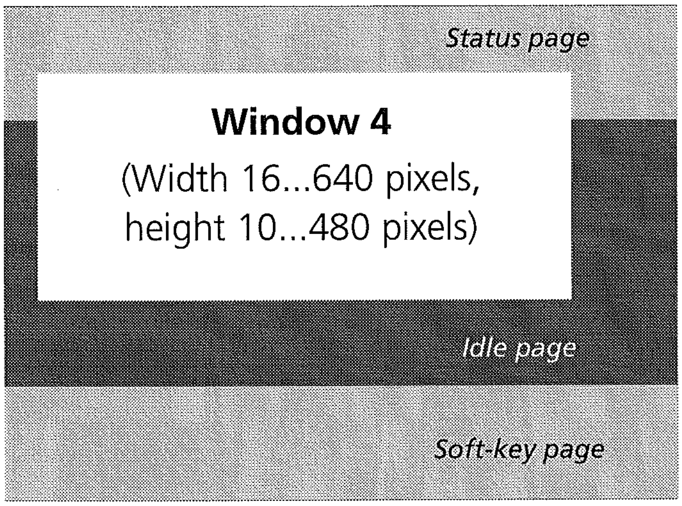
❻ Operating page / menu

Preset values are specified via menus. Each menu is assigned an operating page where all dynamic and static elements are positioned. Window 4 of PCS*maxi* contains 255 menus.

Window 4 covers windows 1 to 3. Positioning of window 4 on the display is effected pixel by pixel (vertical direction) and in steps of 16 pixels (horizontal direction).

Each operating page may contain up to 64 variables which can be edited (preset value) or are only displayed (actual value), depending on the access class.

The access class assignment is specified in the variable definition. The access class in data word 17 and the access class of the interconnected variables determines whether the variable is only displayed or whether it can also be edited.



Page description of an operating page with static text. OP_PAGE,1 specifies the number of the operating page, followed by width and height indication of window 4.

```
            Menu number
            I     Width of the operating page in pixels
            I     I     Height of the operating page in pixels
            I     I     I
```

OP_PAGE, 1, (320, 185)

INS_ST_TEXT, (20, 18), FONT (1)
LANGUAGE (1), "Start"

OP_PAGE_END

Interconnection of the operating page defined above in a menu: The designation Example Menu1 is assigned to menu 1. The value of 1 (after NODE) indicates the operating page to be interconnected. (192, 180) specifies the position of the menu (window 4) - lower left corner. The menu position is obtained by the positioning help (function key F6). The cursor must be positioned on the NODE line).

MENU, Example menu, 1
NODE, 1, (192, 180)

❼ Help page

Help pages support the operator by displaying detailed information about idle pages, messages, operating pages and recipes.

If required, DEFAULT help pages may be parameterized. For each priority (idle page, info, warning, fault and operating page), one help page is then available.

Furthermore, a specific help page may be defined for each idle, message and operating page. This help page is displayed alternatively instead of a DEFAULT help page.

The help page is always assigned to the currently highest priority.

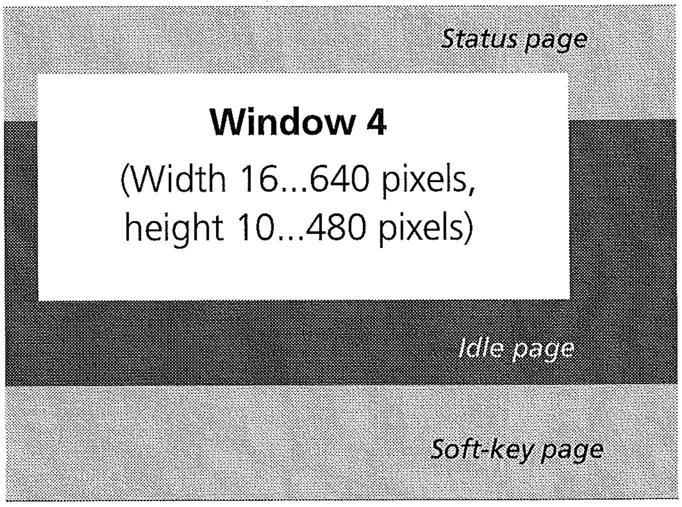
- idle page (lowest priority) - REPORT
- info - INFO
- warning WARNING
- fault - FAULT
- operating page - OP
- recipe (highest priority)

Help pages are displayed in window 4. Windows 1 to 3 are not visible.

To display a help page for an operating page, the menu is closed and the corresponding help page is displayed in window 4.

Positioning of window 4 on the display is effected pixel by pixel (vertical direction) and in steps of 16 pixels (horizontal direction).

All dynamic and static elements may be interconnected. Up to 64 dynamic elements (variables) may be displayed on a help page.



Description of a DEFAULT help page: The elements (dynamic and static) are specified on the page. 64 and 336 indicate the position of the lower left corner of the help page. 512 indicates the width and 170 the height. Size and position are obtained by the positioning help (function key F6). The cursor must be positioned on the HLP_PAGE line).

HLP_PAGE, DEFAULT, REPORT, (64, 336), (512, 170)
INS_ST_TEXT, (99, 86), FONT (1)
LANGUAGE (1), "DEFAULT help page for idle pages 0...255"
HLP_PAGE_END

Page description of a help page 1: Inside, the elements (dynamic and static) are entred.
HLP_PAGE, 1, REPORT, (64, 221), (448, 86)
INS_ST_TEXT, (21, 37), FONT (1)
LANGUAGE (1), "Help page for idle page 1"
HLP_PAGE_END

❽ Operator printer page

Up to 255 operator printer pages are available to create for example shift logs or job-dependent logs which can be printed out or archived and evaluated with a computer (PC, VME etc.).

Operator printer pages contain **static texts**, **dynamic elements** (internal or external variables) and control sequences (#1 to #8 and #LF) used to influence fonts.

Each operator printer page may contain up to **126 lines** with **132 characters** each (including variables). 8 variables can be positioned in each line. Except for graphic variables (GBIT, GSTRING, GCSTRING and bar diagrams), all **variable types** are admissible.

A DEFAULT operator printer page may be created to indicate operator printer pages not correctly loaded during startup of the programmable controller program. This DEFAULT operator printer page is printed out, if the programmable controller is loading an operator printer page which is not explicitly defined.

Description of an idle page for two languages (LANGUAGEMAX, (2)): The elements (dynamic and static) are specified on the page..

```
OP_PRNPAGE, 1
LANGUAGE (1)
PRNTEXT
    The desired text without limiting quotation marks is typed within PRNTEXT and
    PRNTEXT_END. Within PRNTEXT and PRNTEXT_END no comment lines are
    admissible.
    As default, carriage return ($OD) and line feed ($OA) are automatically
    transmitted to the printer after each line.
PRNTEXT_END
LANGUAGE (2)
PRNTXT
    The desired text for the second language - Language (2) - is specified within
    PRNTEXT and PRNTEXT_END.
    Variables are positioned between at(@)-characters:
    Interface errors @[COM_ERRORS]@% (valid in the range of 0 to 4%!)
PRNTEXT_END
OP_PRNPAGE_END
```

Variables are enclosed by @VAR@!

If the internal variable [LANGUAGE] is set to instance 0 (first language), the following text is output on the printer for the operator printer page defined above:

```
The desired text without limiting quotation marks is typed within PRNTEXT and
PRNTEXT_END. Within PRNTEXT and PRNTEXT_END no comment lines are
admissible.
As default, carriage return ($OD) and line feed ($OA) are automatically
transmitted to the printer after each line.
```

If [LANGUAGE] is set to instance 1 (second language), the following is output on the printer:

```
The desired text for the second language - Language (2) - is specified within
PRNTEXT and PRNTEXT_END.
Variables are positioned between at(@)-characters:
Interface errors 1% (valid in the range of 0 to 4%!)
```

Printer control sequences (#1 ... #8 and LF) simplify printer operation and enable the control parameters page feed, line feed, bold characters on/off, italics on/off and underline on/off to be specified. For Epson-compatible printers, the following sequences can be defined (before the operator printer pages):

Printer control sequences (#1 ... #8 and LF) simplify printer operation.

Tab
PRNCONTROL, #1, "$09"

Escape
PRNCONTROL, #2, "$1b"

Carriage return and line feed
PRNCONTROL, #3, "$0d$0a"

Form feed
PRNCONTROL, #4, "$0c"

Bold on
PRNCONTROL, #5, "$1b$45"

Bold off
PRNCONTROL, #6, "$1b$46"

Underline on
PRNCONTROL, #7, "$1b$2d$31"

Underline off
PRNCONTROL, #8, "$1b$2d$30"

#LF is sent after each line. The default contents of #LF is "$0d$0a" (carriage return and line feed).
PRNCONTROL, #LF, "$0d$0a"

Printer control sequences (#1 ... #8 and LF) may be up to 80 characters long.

Printer sequences can also be used to abreviate frequently used text strings.

To create tables with more than 8 variables per line, the default setting *carriage return* and *line feed* „OdOa“ which is automatically inserted after each line can be deleted with

PRNCONTROL, #LF, ""

Line feeds are then inserted manually with #3:

```
OP_PRNPAGE, 1
LANGUAGE (1)
PRNTEXT
More than 8 variables in a line: #3
@V01@ @V02@ @V03@ @V04@ @V05@ @V06@ @V07@ @V08@
@V09@ @V10@ @V11@ @V12@ @V13@ @V14@ @V15@ @V16@#3
#3End of the table.#3#4
PRNTEXT_END
```

All characters which cannot directly be entered with the editor but which must appear in the printout (range $00 to $1f and the command characters #, $ and @) can be inserted in the hexadecimal format:

#	*is*	*$23*
$	*is*	*$24*
@	*is*	*$40*

The sequence #LF which is automatically sent at the end of each line must then be reset to the default value for all following operator printer pages:
PRNCONTROL, #LF, "$0d$0a"

The following is output on the printer:

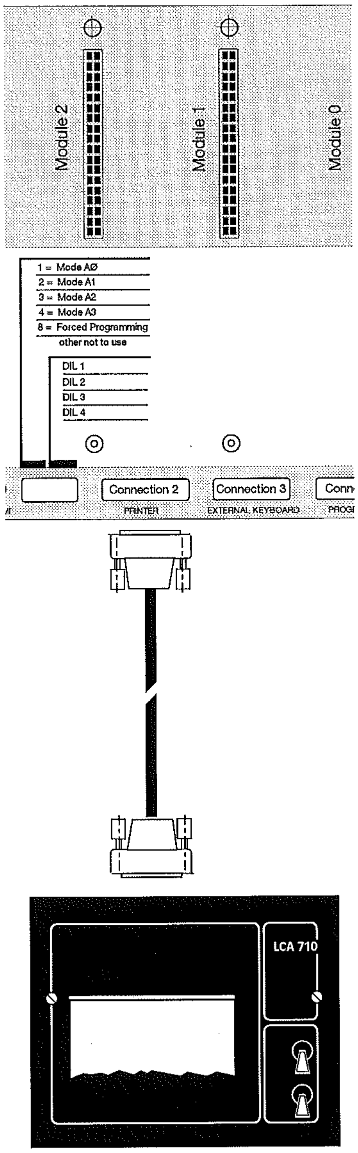
```
More than 8 variables in a line:
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8

 End of the table.
```

Printer interface

PCS*maxi* and the printer are connected via the serial interface RS 232.

Possible interface settings:

Baud rate [PRN_BAUD]:	300, 600, 1200, 2400, 4800, 9600 (default), 19200
Data bits [PRN_DBITS]:	7, 8 (default)
Parity [PRN_PARITY]:	NONE (default), ODD, EVEN
Protocol [PRN_PROT]:	- RTS/CTS (default) - XON/XOFF - XON/XOFF CONTINOUSLY
Stop bits [PRN_SBITS]:	1 (default), 2



Other settings are possible by either overwriting the internal variable located directly after the global declarations in the data record description or by integrating the internal variables in menus. The selected interface settings are stored in the battery-backed RAM of the PCS.

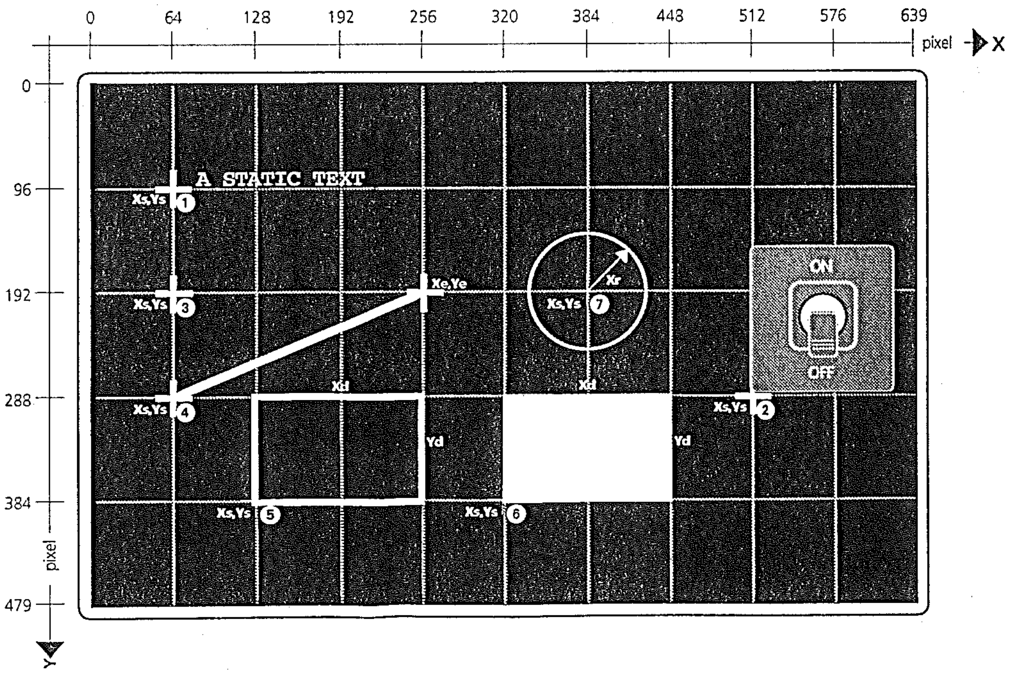
With the **RTS/CTS** protocol the printout is stopped until the printer is ready or a printer is connected. With the **XON/XOFF** setting, printing continues until XOFF is received. If no printer is connected, the print data do not reach their destination. If **XON/XOFF CONTINOUSLY** is set, XON from the printer is waited for before starting the printout. Printing continues until XOFF is received. If the connected printer is not ready, the printout is stopped.

Printing requires hardware version PG9000.207 or later. This is displayed, if the rotary switch is set to 8 and the power supply is switched *off and on* again.

15 Static elements

The PCS*maxi* offers various static elements which can be postioned on the pages:

❶	Static text	*INS_TEXT,(Xs,Ys),Font (1...8)* *INS_TEXT,(64,96),Font (2)*
❷	Static bitmap (images, logos)	*INS_ST_BITMAP,(Xs,Ys)* *INS_ST_BITMAP,(512,288)*
❸	Pixel	*INS_PIXEL,(Xs,Ys)* *INS_PIXEL,(64,192)*
❹	Lines	*INS_LINE,(Xs,Ys),(Xe,Ye)* *INS_LINE,(64,288),(256,192)*
❺	Rectangles	*INS_RECT,(Xs,Ys),(Xd,Yd)* *INS_RECT,(128,384),(128,96)*
❻	Filled rectangles	*INS_FILLRECT,(Xs,Ys),(Xd,Yd)* *INS_FILLRECT,(320,384),(128,96)*
❼	Circles	*INS_CIRCLE,(Xs,Ys),Xr* *INS_LINE,(384,192),50*



❶ Static text

A static text is used in each page description. Within page descriptions, any number of static texts is allowed. One line is requested for each language specified by the global definition LANGUAGEMAX (2).

If, in the global definitions, another character set has been selected for one of the languages with

```
CHARSET (1), 1, 1
CHARSET (2), 2, 2
```

this character set is displayed with the corresponding language.

Insert the static text into a page description. The coordinate Xs=64, Ys=96 (beginning of the text in the display) is determined with the positioning help (function key <F6>).

```
INS_ST_TEXT, (64, 96), FONT (2)
LANGUAGE (1), "Text for language 1"
LANUGAGE (2), "Text for language 2"
```

❷ Static bitmap

Static bitmaps are first of all used to insert logos, background images etc. which can be created under Windows with tools such as Paintbrush, CorelDraw, Designer and others. Scanned images or converted drawings can also be inserted, e.g. from AutoCAD.

Within a page description, any number of static bitmaps is possible.

PCS 9092 enables BMP files with two colors and a size up to 640x480 pixels to be processed.

For conversion purposes, the floppy disk contains different shareware tools used for color reductions, scalings and BMP conversions (Installation is effected with WINTOOLS.BAT).

Insert the static bitmap into a page description. The coordinate Xs=512, Ys=288 is determined with the positioning help (function key <F6>). Before doing this, the bitmap file and the rectangular window are selected with the bitmap editing function (function key F5) and by positioning the cursor into the ICON line.

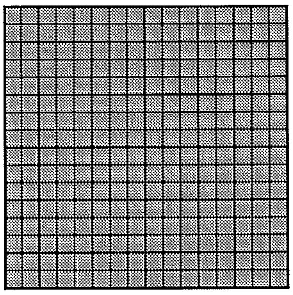
```
INS_ST_BITMAP, (512,288)
ICON, "c:\pcs9092\beispiel\pcs9000.bmp", (0, 399, 640, 480)
```

❸ Pixel

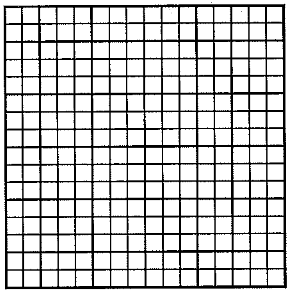
Page descriptions may contain any number of individual pixels.

Insert pixels into a page description. The coordinate Xs=64, Ys=192 is determined with the positioning help (function key <F6>).

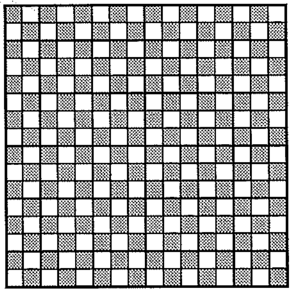
PATTERN (1)



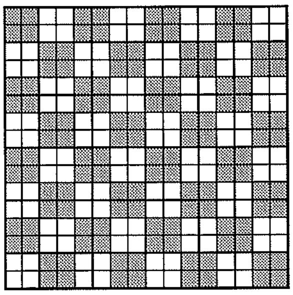
PATTERN (2)



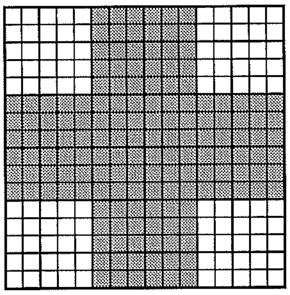
PATTERN (3)



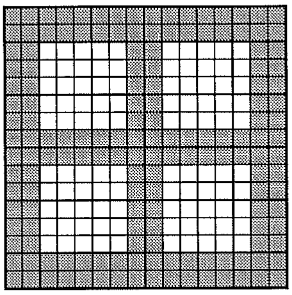
PATTERN (4)



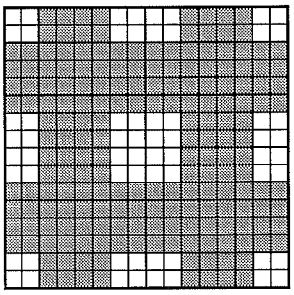
PATTERN (5)



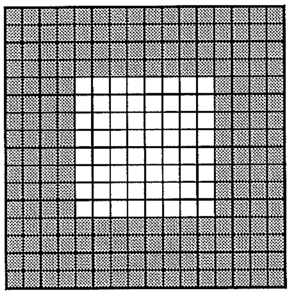
PATTERN (6)



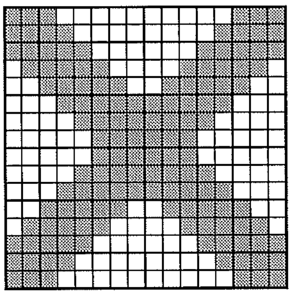
PATTERN (7)



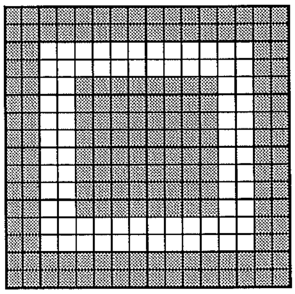
PATTERN (8)



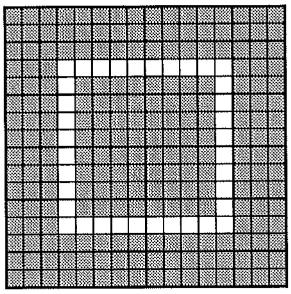
PATTERN (9)



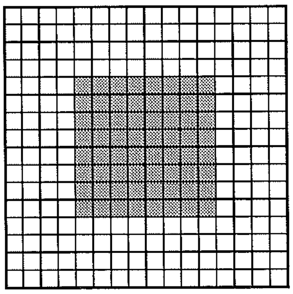
PATTERN (10)



PATTERN (11)



PATTERN (12)



❹ Line

Page descriptions may contain any number of lines. Lines have a width of 1 pixel and any patterns can be assigned to lines.

Insert a static line into a page description. The start and end point of the line are determined by means of the positioning help (function key <F6>).

INS_LINE, (64, 288), (256, 192)
PATTERN (1)

Alternatively, a BMP file with a window size of 16x16 pixels and a user-specified fill pattern can be created (function key F5 in the PATTERN line).

INS_LINE, (34, 45), (67, 78)
PATTERN, "patt1.bmp", (0, 15, 16, 16)

In the PCS*maxi*, 12 predefined fill patterns (PATTERN (1) to PATTERN (12)) are available. These pattern generally have a size of 16x16 pixels. For lines, the created fill patterns with a size of 16x16 pixels are used in the following way:

240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255
224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

End of the line

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27

❺ Rectangle

Any number of rectangles can be used within page descriptions. Rectangles generally have a size of 1 Pixel.

Insert the rectangle into a page description. The coordinates (always lower left corner of the rectangle) for the rectangle position on the page Xs=128, Ys=384 and the size Xd=128, Yd=96 are determined with the positioning help (function key <F6>).

INS_RECT, (128,384), (128, 96)

❻ Filled rectangle

Page descriptions may contain any number of filled rectangles.

Insert the rectangle into a page description. The coordinates Xs=255, Ys=155 and the size Xd=20, Yd=30 are determined with the positioning help (function key <F6>). The internal fill pattern 3 is enabled.

NS_FILLRECT, (320,384), (128, 96)
PATTERN (3)

The fill pattern (window size of 16x16 pixels) can also be loaded from a BMP file (function key F5 in the PATTERN line).

INS_FILLRECT, (255,155), (20, 30)
PATTERN, "patt1.bmp", (0, 15, 16, 16)

❼ Circle

Page descriptions may contain any number of circles. Circles are drawn with a line width of 1 pixel.

Insert the circle into a page description. The coordinates Xs=255 and Ys=155 indicate the center point of the circle and Xr=20 the radius. The center point and the radius are determined with the positioning help (function key <F6>).

INS_CIRCLE, (384,192), 20

Modifying manufacturing parameters and functions (i.e. variable elements) in a fast and specific way is essential to attain a high machine or plant flexibility.

The enormous projection flexibility of PCS*maxi* is based on extremely user-friendly variable editing and processing functions. In the PCS, a distinction is made between external and internal variables (the internal variables are generated by the PCS itself).

Access class, menu number

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
128	64	32	16	8	4	2	1	128	64	32	16	8	4	2	1
Access class								Menu number							

DW 17

```
Command,     Name,           Data word,   Access class,   Format,   Bit pos
VAR,         EXHAUSTION,     33,          8,              BIT,      0
STATE TEXT, 0
LANGUAGE (1), " EXHAUSTION OFF"
LANGUAGE (2), " OFF"
STATE TEXT, 1
LANGUAGE (1), " EXHAUSTION ON"
LANGUAGE (2), " ON"
```

Variables used as dynamic elements are actual or preset values:

ACTUAL: *The variable is an actual value which is only displayed by the PCS.*

PRESET: *The variable is a preset value which is displayed by the PCS and can be modified in the menu.*

The **access class** for variables enables up to 255 different access rights to be assigned. An access class is allocated to each variable:

Access class 0 are actual values.
Access classes 1...255 are preset values.

Changing a variable in a menu is only possible, if the corresponding right has been granted via the access class.

Example: *The access class for a bit variable is 8 and access class 10 is specified in the data word 17 (bits 8...15): The bit variable EXHAUSTION can thus be modified.*

Generally, variables for which an access class lower than or equal to the access class specified in data word 17 is defined, can be modified. Variables with higher access rights cannot be modified.

Internal variables

Parameters can be assigned to these internal variables (replacement of default value directly after the global definitions)

Format	*Function*	*Format*	*Length*	*Access class*
[CLK_CADR]	Clock address in the programmable controller	BIN-2	3	0
[CLK_CBIT]	Bit position for clock transfer locking in DW 13	BIN-2	2	0
[CLK_DADR]	Programmable controller address for date	BIN-2	3	0
[CLK_DBIT]	Bit position for date transfer locking in DW 13	BIN-2	2	0
[CLK_DOFW_ACT]	Day of the week	STRING	any	0
[CLK_DOFW_NOM]	Day of the week	STRING	any	1
[LANGUAGE]	Language selection	STRING	any	1
[OPPRT_DW]	Operator printer page address in the programmable controller	BIN-2	3	0
[OPPRT_SBIT]	Bit position operator printer page transfer locking in DW 13	BIT	2	0
[PRN_BAUD]	Baud rate internal RS 232 printer interface	STRING	any	1
[PRN_DBITS]	Data bits internal RS 232 printer interface	BIN-2	1	1
[PRN_PARITY]	Parity internal RS 232 printer interface	STRING	any	1
[PRN_PROT]	Protocol internal RS 232 printer interface	STRING	any	1
[PRN_SBITS]	Start bits internal RS 232 printer interface	BIN-2	1	1

Internal variables providing information in the page descriptions

[CLK_COUNT_1]	Counter 1	BIN-2	10	1
[CLK_COUNT_2]	Counter 2	BIN-2	10	1
[CLK_COUNT_3]	Counter 3	BIN-2	10	1
[CLK_COUNT_HN]	Elapsed-hour meter	BIN-2	10	1
[CLK_COUNT_H]	Elapsed-hour meter	BIN-2	10	0
[CLK_DATE_ACT]	Complete date	-	8	0
[CLK_DAY]	Day	BIN-2	2	1
[CLK_DOFW_ACT]	Day of the week	STRING	any	0
[CLK_DOFW_NOM]	Day of the week	STRING	any	1
[CLK_HOUR]	Hour	BIN-2	2	1
[CLK_MINUTE]	Minute	BIN-2	2	1
[CLK_MONTH]	Month	BIN-2	2	1
[CLK_SECOND]	Seconde	BIN-2	2	1
[CLK_TIME_ACT]	Complete clock		8	0
[CLK_YEAR]	Year	BIN-2	2	1
[LANGUAGE]	Language selection	STRING	any	1
[MSG_FAULT]	Number of active faults	BIN-2	4	0
[MSG_INFO]	Number of active infos	BIN-2	4	0
[MSG_TCOME]	Complete date and time - message received	-	17	0
[MSG_TGONE]	Complete date and time - message gone	-	17	0
[MSG_TQUIT]	Complete date and time - message acknowledged (CLR key)	-	17	0
[MSG_TXTNR]	Number of the message page	BIN-2	4	0
[MSG_WARN]	Number of active warnings	BIN-2	4	0
[PRN_BAUD]	Baud rate internal RS 232 printer interface	STRING	any	1
[PRN_DBITS]	Data bits internal RS 232 printer interface	BIN-2	1	1
[PRN_PARITY]	Parity internal RS 232 printer interface	STRING	any	1
[PRN_PROT]	Protocol internal RS 232 printer interface	STRING	any	1
[PRN_SBITS]	Start bits internal RS 232 printer interface	BIN-2	1	1
[STRT_KEYCODE]	Action group (0...255) of the most recently pressed key	BIN-2	3	0
[STRT_KEYGRP]	Action category (1...3) of the most recently pressed key	BIN-2	1	0

COM variables are driver dependent. They change in respect to the driver used (more information can be found in the driver manual PCS91.XXX)

External variables

Data words 25...255 are available in the PCS*maxi* for the contents of the external or internal variables. For the external variables, the following variable formats are used:

❶ BIT and STRING variable

❷ BCD and BIN variable

❸ Word variable

❹ ASCII variable

❺ Bargraph

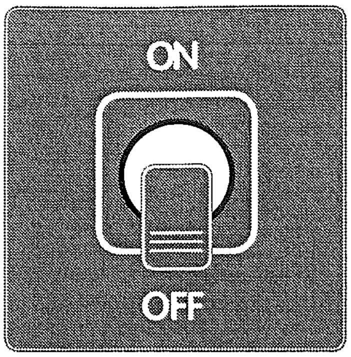
	Format	*Size*
❶	*BIT variable*	*max. length 80 characters*
❶	*STRING variable*	*max. length 80 characters*
❶	*CSTRING variable*	*max. length 80 characters*
❶	*GBIT variable*	*max. 640 x 480 pixels*
❶	*GSTRING variable*	*max. 640 x 480 pixels*
❶	*GCSTRING variable*	*max. 640 x 480 pixels*
❷	*BCD-1 variable*	*max. length 4 digits*
❷	*BCD0-1 variable*	*max. length 4 digits*
❷	*BCD-2 variable*	*max. length 8 digits*
❷	*BCD0-2 variable*	*max. length 8 digits*
❷	*BIN-1, BIN-A variable*	*max. length 16 bit/11 digits*
❷	*BIN0-1, BIN0-A variable*	*max. length 16 bit/11 digits*
❷	*BIN-2, BIN-B variable*	*max. length 32 bit/11 digits*
❷	*BIN0-2, BIN0-B variable*	*max. length 32 bit/11 digits*
❷	*VBIN-1, VBIN-A variable*	*max. length 16 bit/12 digits*
❷	*VBIN0-1, VBIN0-A variable*	*max. length 16 bit/12 digits*
❷	*VBIN-2, VBIN-B variable*	*max. length 32 bit/12 digits*
❷	*VBIN0-2, VBIN0-B variable*	*max. length 32 bit/12 digits*
❸	*WORD variable*	*length 17 digits*
❹	*ASCII variable*	*max. length 16 characters*
❺	*PHBAR*	*max. 640 x 480 pixels*
❺	*NHBAR*	*max. 640 x 480 pixels*
❺	*PVBAR*	*max. 640 x 480 pixels*
❺	*NVBAR*	*max. 640 x 480 pixels*
❺	*SHSYMBAR*	*max. 640 x 480 pixels*
❺	*SVSYMBAR*	*max. 640 x 480 pixels*

BIT variable

BIT variables are used for specifying two different selections. This corresponds to an ON/OFF switch.

Each switch position is assigned an instance (text) which appears on the display. Each BIT variable is assigned a bit. This means that a word may contain up to 16 different BIT variables or switches.

The +/- key or the INC/DEC keys are used to select the instance or switch position. The bit is assigned the value of the instance. The value logical 0 is allocated to the first and logical 1 to the second instance.



Switch

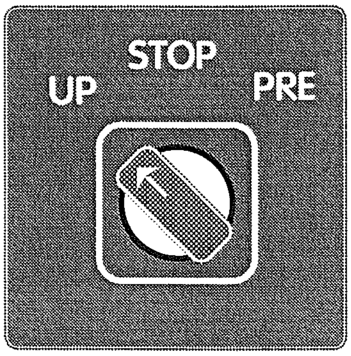
```
Command,  Name,         Data word,  Access class,           Format,  Bitpos
VAR,      EXHAUSTION,   33,         1,                      BIT,     0
STATE TEXT, 0                       Instance for logical 0
LANGUAGE (1), " EXHAUSTION OFF"     Language 1
LANGUAGE (2), " OFF"                Language 2
STATE TEXT, 1                       Instance for logical 1
LANGUAGE (1), " EXHAUSTION ON"      Language 1
LANGUAGE (2), " ON"                 Language 2
Access class: 0 = actual value, 1...255 = preset value
```

STRING variable

A STRING variable is used for an entry specifying more than two conditions. This corresponds to the function of a selector switch.

If STRING variables are used, each switch position is assigned an instance (text) which appears on the display. Each STRING variable is assigned a word with up to 256 switch positions. The switch position is allocated to the LOW byte of the word.

The INC/DEC keys are used to select the instances or switch positions. To confirm the selection, the ENTER key can for example be used.



Selector switch

```
Command,  Name,         Data word,  Access class,  Format
VAR,      SHIFT,        43,         0,             STRING
STATE TEXT, 0                       Instance 0
LANGUAGE (1), " EXHAUSTION OFF"     Language 1
LANGUAGE (2), " OFF"                Language 2
STATE TEXT, 1                       Instance 1
LANGUAGE (1), " EXHAUSTION ON"      Language 1
LANGUAGE (2), " ON"                 Language 2
STATE TEXT, 2                       Instance 2
LANGUAGE (1), " EXHAUSTION READY"   Language 1
LANGUAGE (2), " STANDBY"            Language 2
Access class: 0 = actual value, 1...255 = preset value
```

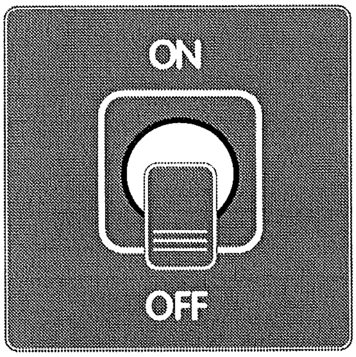
CSTRING variable

The CSTRING variable corresponds to the STRING variable. The setting is enabled immediately after pressing the INC/DEC keys (without ENTER).

❶ GBIT variable

Each switch condition is assigned an instance (**graphics**) which appears on the display. Each BIT variable is assigned a bit. This means that a word may contain up to 16 different BIT variables or switches.

The +/- key or the INC/DEC keys are used to select the instance or switch position. The bit is assigned the value of the instance. The value logical 0 is allocated to the first and logical 1 to the second instance.



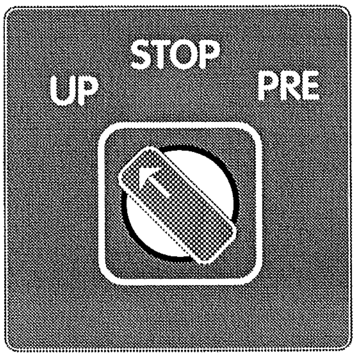
Switch

```
Command,   Name,          Data word,   Access class,   Format,   Bit pos
VAR,       Switch,        33,          10,             GBIT,     1
STATE ICON, 0                          Instance for logical 0
ICON, "D:pattern1.bmp", (10,60,40,40)  display BITMAP ⏻
STATE ICON, 1                          Instance for logical 1
ICON, "D:pattern2.bmp", (10,60,40,40)  display BITMAP ⏽
Command,              BITMAP file,     Rectangular section from the BITMAP file
Access class: 0 = actual value, 1...255 = preset value
```

❶ GSTRING variable

If STRING variables are used, each switch position is assigned an instance (**graphics**) which appears on the display. Each STRING variable is assigned a word with up to 256 switch positions. The switch position is allocated to the LOW byte of the word.

The INC/DEC keys are used to select the instances or switch positions. To confirm the selection, the ENTER key can for example be used.



Selector switch

```
Command,   Name,          Data word,   Access class,   Format
VAR,       Switch,        40,          1,              GSTRING
STATE ICON, 0                          Instance 0
ICON, "D:pattern10.bmp", (10,30,85,85) display BITMAP ▲
STATE ICON, 1                          Instance 1
ICON, "D:pattern11.bmp", (10,30,85,85) display BITMAP ▶0◀
STATE ICON, 2                          Instance 2
ICON, "D:pattern12.bmp", (10,30,85,85) display BITMAP ▼
Command,              BITMAP file,     Rectangular section from the BITMAP file
Access class: 0 = actual value, 1...255 = preset value
```

❶ GCSTRING-Variable

The GCSTRING variable corresponds to the GSTRING variable. The setting is enabled immediately after pressing the INC/DEC keys (without ENTER).

BITMAP graphics variables can be created with any graphics programs or loaded from our symbols library. The PCS 9092 program processes BMP files with up to 640 x 480 pixels.

❷ BCD variable: BCD-1, BCD-2, BCD0-1, BCD0-2

The BCD variable corresponds to a BCD encoding switch (preset value) or to a BCD coded digital display (actual value). The 4-digit variable BCD-1 is allocated to a word and the 8-digit variable BCD-2 to two consecutive words 32 bit (W n, W n+1). BCD variables are displayed without leading zeros. Example for a BCD-2 actual value: 4 2567.

The 4-digit variable BCD0-1 is allocated to a word and the 8-digit variable BCD0-2 to two consecutive words 32 bit (W n, W n+1). BCD0 variables are displayed with leading zeros. Example for a BCD0-2 actual value: 0004 2567.

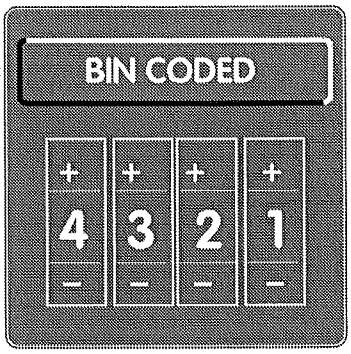
Each BCD variable can be limited by a min/max value.

The preset value 8500 is entered via the numeric keyboard of the PCS and is allocated to the following word in the BCD format, for example by pressing the ENTER key.

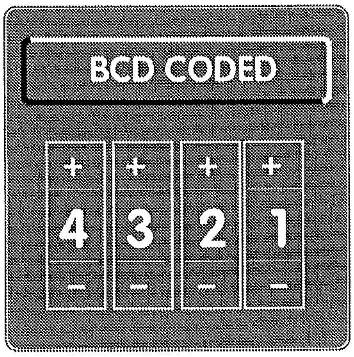
```
Word 30 =  1000  0101  0000  0000

              8     5     0     0
```

Command,	*Name,*	*Data word,*	*Access class,*	*Format,*	*Digit,*	*Min value,*	*Max value*
VAR,	*Speed,*	*30,*	*0,*	*BCD-1,*	*4,*	*0,*	*9999*

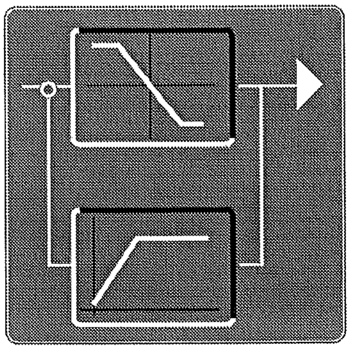
Access class: 0 = actual value, 1...255 = preset value



BIN preset value input



BCD preset value input



Limit value scaling

❷ BINARY variable: BIN-1, BIN-2, BIN-A, BIN-B, VBIN-1, VBIN-2, VBIN-A, VBIN-B, BIN0-1, BIN0-2, BIN0-A, BIN0-B, VBIN0-1, VBIN0-2, VBIN0-A, VBIN0-B

The BIN variable corresponds to a BINary encoding switch (preset value) or to a digital display (actual value). The scaleable 16-bit variables (BIN-1 to VBIN-A) are allocated to a word and the 32-bit variables (BIN-2 to VBIN-B) to two consecutive words (W n, W n+1).

The variables BIN-1 to VBIN-2 differ from BIN-A to VBIN-B only in the way numbers are entered. VBIN-1 to VBIN-B take into account the sign. So the possible input range is reduced to the half. Each BIN variable can be limited by a min/max value. Furthermore, in case of the BIN-1 to VBIN-A variable, the value range of the programmable controller can be converted into the value range of the PCS (scaling).

Examples:

The temperature of the motor brake is adjustable between 0°C and 70°C. The scaled 16-bit preset value ist written into word W 45 (with min/max limit).

Command,	*Name,*	*Data word,*	*Access class,*	*Format,*	*Pre dec. point,*	*Post dec. point,*	*min PCS,*	*max PCS,*	*min prog.contr ,*	*max prog. contr*
VAR,	*Speed,*	*45,*	*100,*	*BIN-1,*	*2,*	*0,*	*0,*	*70,*	*0,*	*1024*

Access class: 0 = actual value, 1...255 = preset value

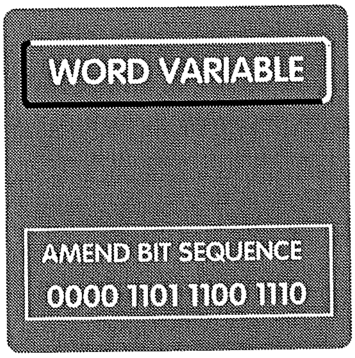
The window height is adjustable between 750 mm and 1500 mm. The 32-bit preset value ist written into word W 41 + 42 (with min/max limit). The more significant part is allocated to W 41 and the less significant part to W 42.

Command,	*Name,*	*Data word,*	*Access class,*	*Format,*	*Pre dec. point,*	*Post dec. point,*	*Min value,*	*Max value*
VAR,	*Windows height,*	*41,*	*100,*	*BIN-2,*	*4,*	*0,*	*750,*	*1500*

Access class: 0 = actual value, 1...255 = preset value

❸ WORD variable

The WORD variable is ideal for the service, since it displays the bit pattern of a word on the PCS. Changing the bit pattern with the PCS is easy, if the WORD variable is defined as preset variable.



WORD VARIABLE

AMEND BIT SEQUENCE
0000 1101 1100 1110

Change data word/flag

Example: Word 33 is displayed on the PCS and modified:

The preset value WORD variable is modified via a menu.

Display line 1 *BITPATTERN OF WORD 31*
Display line 2 *00000000 00000101*

The value of the WORD variable is changed via the key 0 or 1. The INC key is used to move the pointer one digit to the right and the DEC key to move it one digit to the left.

Display line 1 *BITPATTERN OF WORD 31*
Display line 2 *11110000 11000000*

Press the ENTER key to allocate the new value to the word.

Command,	*Name,*	*Data word,*	*Access class,*	*Format*
VAR,	*Bitpattern,*	*31,*	*10,*	*WORD*

Access class: 0 = actual value, 1...255 = preset value

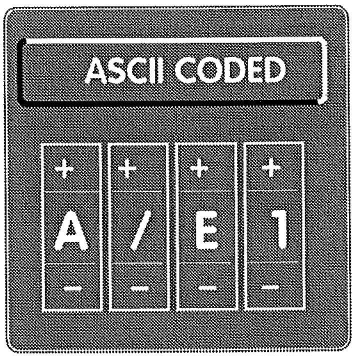
❹ ASCII variable

The ASCII variable is used, if an alphanumeric preset value (item number, names etc.) is required.

Example: A 12-digit version number (41-BN-890-SB) must be entered:

Since two ASCII characters are assigned to a word, 6 words must be reserved for a 12-digit version number. In the following example, the words 56...61 are used for this.

The ASCII variable (preset value) is entered via a menu.



ASCII preset value input

Display line 1	*ENTER THE VERSION:*
Display line 2	■■■■■■■■■■■■

When the menu is called, the value 0 is allocated to the words 56...61. For this value (00), the PCS character table sets this characters to n (all dots light). If a key is pressed, these characters are replaced by a question mark (?).

Display line 1	*ENTER THE VERSION:*
Display line 2	*????????????*

Each question mark (?) can be paged up with the A->Z key and paged down with the Z ->A key and thus be replaced by an alphanumeric character. The key -> is used to move the cursor one digit to the right.

Display line 1	*ENTER THE VERSION:*
Display line 2	*41-BN-890-SB*

If the characters have been entered completely and correctly, the value is assigned to the word by pressing for example the ENTER key. In this case, the data words have the following content:

Word #	*Content*	*ASCII character*
W56	*34 31*	*4 1*
W57	*2D 42*	*- B*
W58	*4E 2D*	*N -*
W59	*38 39*	*8 9*
W60	*30 2D*	*0 -*
W61	*53 42*	*S B*

Overview of the keys used for ASCII variables

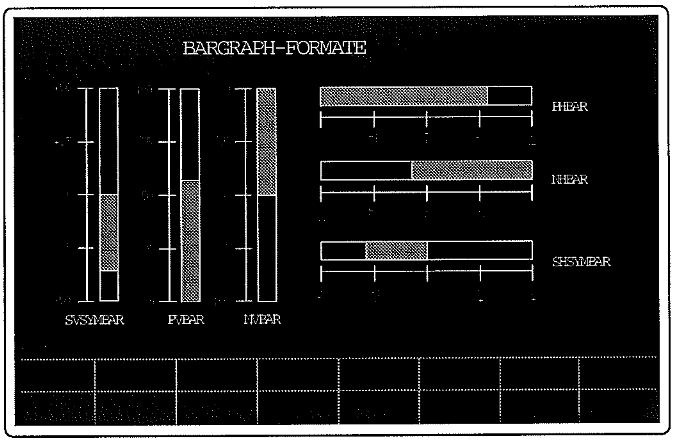
A->Z key	page up within the alphabet
Z->A key	page down within the alphabet
-> key	move the cursor one digit to the right.
ENTER key	write the ASCII characters into the data words in hex code
CLR key	delete the entry and display the old value

Command,	*Name,*	*Data word,*	*Access class,*	*Format,*	*# data words*
VAR,	*Part,*	*30,*	*20,*	*ASCII,*	*8*

Access class: 0 = actual value, 1...255 = preset value

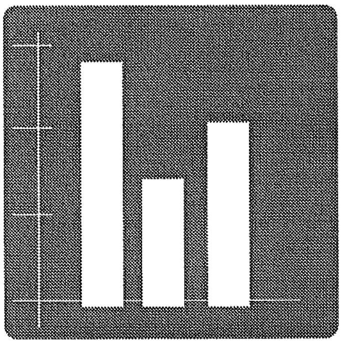
❺ Bargraphs

Bargraphs are used for graphical representation of numeric values. Each bar moves between an upper and lower limit. If the limits are exceeded, the fill pattern which can be defined as required changes. The bargraph type and size can be customized.

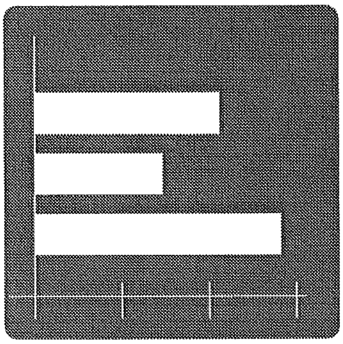


3 bargraph formats (horizontal, vertical, symmetric) are available:

PHBAR	Positive horizonal bar, expands to the right
NHBAR	Negative horizonal bar, expands to the left
PVBAR	Positive vertical bar, expands upwards
NVBAR	Negative vertical bar, expands downwards
SHSYMBAR	Horizontal symmetric bar, expands to the left and to the right
SVSYMBAR	Vertical symmetric bar, expands upwards and downwards



Vertical bargraph



Horizontal bargraph

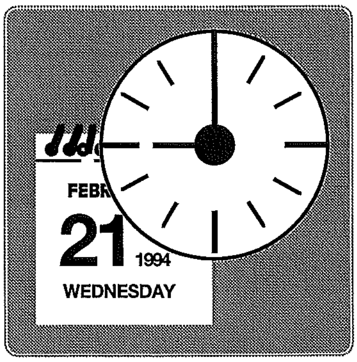
```
Command,  Name,          Data word,  Access class,  Format,   Bar(width/height),  Min prog.contr,  Max prog. contr
VAR,      Temperature,   60,         0,             PHBAR,    (100,10),           10,              100
FILL,     "D:pattern 20.bmp", (10,10,16,16)     pattern for filled area
EMPTY,    "D:pattern 21.bmp", (100,10,16,16)    pattern for empty area
INVALID,  "D:pattern 22.bmp", (200,10,16,16)    pattern for area exceedings
Command,  BITMAP file,        Rectangular section from the BITMAP file

Command,  Name,          Data word,  Access class,  Format,      Bar(width/height),  neg/pos directed
VAR,      Statistic A,   64,         0,             SHSYMBAR,    (80,15),            100
FILL,     "D:pattern 20.bmp", (300,10,16,16)    pattern for filled area
EMPTY,    "D:pattern 21.bmp", (10,10,16,16)     pattern for empty area
INVALID,  "D:pattern 22.bmp", (100,10,16,16)    pattern for area exceedings
Command,  BITMAP file,        Rectangular section from the BITMAP file
```

Access class: bars only for ACTUAL VALUES -> therefore 0.

Date and time

PCS*maxi* is equipped with a real time clock which can be synchronized with the programmable controller. Date and time can be displayed (actual value) and entered (preset value) via internal variables.

In the transfer area, several data words enable access to year, month, day of the week, hour, minutes and seconds. The transfer area configuration of the programmable controller can be selected as required within the limits (word 25 to word 255).



Date and time

	Command,	*Name,*	*Data word*
Example:	*INTVAR,*	*[CLK_DADR],*	*30*

allocates DW 30 and DW 31 for date data.

INTVAR, *[CLK_DBIT],* *15*
allocates bit 15 in DW 13 for the locking bit for date data.
If the internal variable is not parameterized, the date cannot be read out and no data words are assigned in the data block (DB).

	Command	*Name,*	*Data word*
Example:	*INTVAR,*	*[CLK_CADR],*	*40*

allocates DW 30 and DW 31 for day of the week and time data.

INTVAR, *[CLK_CBIT],* *14*
allocates bit 14 in DW 13 for the locking bit for day of the week and time data.
If the internal variable is not parameterized, the day of the week and the time cannot be read out and no data words are assigned in the data block (DB).

Standard installation in the directory C:\PCS9092

Important information concerning installation and use of the PCS 9092 are contained in the file README.EN on the floppy disk and (after the installation) in the directory C:\PCS9092.

The installation requires a disk capacity of 4.1 MBytes, MS-DOS 3.3, DR-DOS 4.0 or higher.

For standard installation, please insert the PCS 9092 floppy disk in the corresponding drive. Change to the selected directory and type INSTALL. This command is used to load the installation program. The directory C:\PCS9092 is now created on your hard disk and all required files are transmitted into this directory.

If Windows 3.1 is installed on your computer, you can use the Windows shareware tools which are also contained on the floppy disk. To install the Windows shareware tools, change again to the floppy drive and type WINTOOLS. The program Paint Shop Pro is then installed in the directory C:\WINDOWS\PSP and the program Graphics Workshop in the directoryC:\WINDOWS\GWS. The versions of these two products are not registered. Please observe the licence agreements listed in the programs.

Installation in another directory or drive

To install the software in another directory or drive, the files

```
PKUNZIP.EXE
PCS9092.ZIP
WINTOOLS.ZIP
```

must be copied from the floppy disk into the desired directory. Change to the desired directory and type in the command

```
"PKUNZIP -o -d  PCS9092" and
"PKUNZIP -o -d  WINTOOLS"
```

All required files are now de-compressed in your directory. Since all files on the floppy disk are compressed, they cannot be used, if you transfer them into your directory with the COPY command. After completing the installation, the three files mentioned above can be deleted.

User memory

In order to use the PCS 9092 program in its full functionality, a user memory of at least 550 kBytes should be available in the PC or programmer. If you enter the DOS command MEM or CHKDSK, the available user memory is displayed. If the available user memory is less than 550 kBytes, delete memory-resident programs and load the operating system into the Upper Memory Blocks, if possible. With MS-DOS 6.0 and later, the program MEMMAKER enables an optimal memory configuration.

Files and their meaning:

ANIMAT	Directory with 3 example programs (see below)
EXAMPLE	Directory with example project
LIB_MECH	Directory with BMP files of the mechanical DIN library (see below)
STEP	Directory with step examples related to this manual
VARIABLN	Directory with examples of all available variable types
S.BAT	Executes the external simulation
KILL.COM	Simulation
PDRIVE.COM	Simulation
PCSD.EXE	Simulation
PCSD.DBS	Simulation
PCSD.MEM	Simulation
PCSD.MSK	Simulation
P.BAT	Start of the PCS 9092
*.E??	Various firmware moduls (START.E02...)
P437_8.PTK	Character set 8x10 Codepage 437
P437_16.PTG	Character set 16x20 Codepage 437
P852_8.PTK	Character set 8x10 Codepage 852
P852_16.PTG	Character set 16x20 Codepage 852
P866_8.PTK	Character set 8x10 Codepage 866
P866_16.PTG	Character set 16x20 Codepage 866
PCS9092.E??	Packed fault texts in different languages
PCS9092.EXE	Programming software PCS 9092
PCS9092.FRM	Default firmware modules for new projects
PCS9092.??	Packed info texts in different languages
PCS9092.H??	Help texts for syntax help to be parametrized by the user
PCS9092.INI	Current configuration of the PCS 9092
*.PCS	Operating project file - can be included in the INCLUDE file (*.INC)
*.INC	individual project section of an operating project

Examples:

It is recommended to install each operating project in a separate sub-directory. Different example projects are located in different sub-directories:

- *Animation demonstration with the PC or the programmer as programmable controller substitution*

The program DEMO9000.EXE is located in the directory ANIMAT. This program and the data record ANIMA480.PCS (for devices with active LCD display) or the data record ANIMA400.PCS (for plasma devices - start PCS 9092 with P /400) enable an animation to be performed with the PC. The data record ANIMA481.PCS contains two languages, two loaded character sets and a printer control. (Operation requires an external memory extension, for example PCS 8110).

Proceed as follows:

1. Compile the desired operating project ANIMA4XX.PCS with the PCS 9092 and transfer it into the PCS9000/PCS9100.

2. Connect the communication interface of the PCS 8010 with the cable PCS 733 and check the rotary switch position at the PCS (switch position 2). Switch the PCS off and on again to read in the switch position again after modification.
3. Start the program with DEMO9000.BAT. Animation is performed via COM 1 as specified in the default settings. The program can also be started manually.

```
Program execution:   DEMO9000    COM   DELAY
                     |           |     |
                     |           |     Demo cycle (depending on the PC default 100)
                     |           Interface 1 - 4 (default 1)
                     Program name
```

Sequence

If you press <SPACE>, the PCS sequence is stopped and the PC displays help information. Simulation continues, if <SPACE> is pressed again.

Function key assignment of PCSmaxi

F3	=	Go to idle page 0.
F4	=	Basic status (remove all parts)
F6	=	Start (machine is started automatically)
F7	=	Stop (machine is stopped). In the stop mode, parts can be moved with F10 and F11.
F9	=	Page through individual idle pages - 0 to 15 (machine image and animation idle page 1)
F15	=	Open and close menu 15 (PCS configuration)
F16	=	Open and close menu 14 (change individual variables)
HLP	=	Display help on the PCS and the PC

- **EXAMPLE.PCS**
  Complete example with different menus for operator guidance in an operating project

- **STEP01.PCS to STEP09.PCS**
  Programming example (step by step) in combination with the PRACTICE manual

- **VARIABLN.PCS**
  Representation of all variable types as actual and preset values in 5 menus.

Library

The complete DIN symbols library MECHANICS is located in the BMP files MECH01.BMP ... MECH09.BMP in the sub-directory LIB_MECH. These files in the BMP format (BMP = bitmap, pixel graphics) can be edited under Windows, for example with Paintbrush. Bitmaps are used in the BMP format in the PCS 9092 for inserting static background images (INS_ST_BITMAP) or in graphic variables (GBIT, GSTRING or GCSTRING) for the individual instances, but also as fill patterns for bar diagrams (PHBAR, NHBAR, PVBAR, NVBAR, SHSYMBAR or SVSYMBAR), filled rectangles (INS_FILLRECT) and as line patterns for lines (INS_LINE).

Graphics Workshop and Paint Shop Pro

Complete machine images or part images can be digitized with scanners and converted into the BMP format with convert programs such as GWS (Graphics Work Shop) or PSP (Paint Shop Pro) which are running under Windows. These images can then be used in the PCS9000/PCS9100. GWS and PSP are distributed as shareware and are not registered before delivery. Detailed information about the licence agreements can be found in the programs.

Circuit diagrams and other diagrams which are stored in the DXF (Data Exchange Format (AutoCAD or AutoSketch)) or HPGL (Hewlett Packard Graphics Language) format can be retrieved under Windows, for example with CorelDraw, Pagemaker, Designer etc., and converted into the BMP format.

Notes concerning operation:

- **Can the PCS 9092 be used under Windows 3.X?**

  Yes. Project planning is possible under Windows in the non-interlaced display mode. Since Windows is characterized by high interrupt latencies, data record transfer into the PCS and simulation can only be performed reliably outside of Windows.

- **Why do internal variables for insertion into page descriptions (e.g. REPORT_PAGE ... REPORT_PAGE_END) do not exist with syntax help?**

  1. The corresponding firmware module was not interconnected (menu item PROJECT/ FIRMWARE MODULES).
  2. In the syntax help, only INTERNAL VARIABLES and not INSERT ELEMENT/INTERNAL VARIABLES was selected.

- **How can variables flash or be inverted?**

  If BIT, STRING or CSTRING variables are interconnected as graphic variables (GBIT, GSTRING or GCSTRING), the programmable controller can change between inverted and non-inverted bitmap by switching the instance. The maximum flash speed depends on the selected communication (PCS-programmable controller) and the scan time of the programmable controller.

- **How are date, time and language configured in the programmable controller?**

  Date, time and language are configured in a menu by means of internal variables. For this task, OP_PAGE 15 is available in the operating project ANIMA480.PCS in the directory ANIMATE and can directly be used for this purpose.

Large operating projects

The editor integrated in the PCS 9092 can be used for files up to 64 kBytes. To realize larger projects and to speed up access to specific page descriptions and elements, it is recommended to create an INCLUDE file for each page description, including variables definition. The extension INS is assigned to INCLUDE files. They are linked to the operating project (PCS file) by the command

INCLUDE, "NAME.INC".

The example projects EXAMPLE.PCS and VARIABLN.PCS on the floppy disk use this technique. To identify the content of the individual pages through the file name, the following system is recommended:

Project:	"PROJECT.PCS"
Idle page (default)	"RPDEF.INC"
Idle page 0	"RP000.INC"
up to	
idle page 255	"RP255.INC"
Soft-key labeling (default)	"SKDEF.INC"
Soft-key labeling 0	"SK000.INC"
Status page	"STATUS.INC"
Heading page	"MSGTITEL.INC"
Message page (default)	"MSGDEF.INC"
Message page 0	"MSG0000.INC"
Operating page1 with menu 1	"OP001.INC"
Help page REPORT (Default)	"HLPRDEF.INC"
Help page REPORT 0	"HLPR000.INC"
Help page INFO (Default)	"HLPIDEF.INC"
Help page INFO 0	"HLP0000.INC"
Help page WARNING (Default)	"HLPWDEF.INC"
Help page WARNING 1	"HLPW0001.INC"
Help page FAULT (Default)	"HLPFDEF.INC"
Help page FAULT 2	"HLPF0002.INC"
Help page OP (Default)	"HLPODEF.INC"
Help page OP 1	"HLPO000.INC"
Operator printer page (default)	"OPPRNDEF.INC"
Operator printer page 15	"OPPRN015.INC"

To speed up positioning and page preview, it is possible to comment out INCLUDE files which are currently not required (insert a leading space in the PCS file before INCLUDE).

Keyboard and mouse button assignment

In the editor, the function keys and functions depend on the **position of the cursor**:

- An element can only be **positioned**, if the cursor is located in the corresponding line (for example INS_ST_TEXT for static text, HLP_PAGE for help pages, NODE for operating pages in menus).
- The **page preview** can only be started, if the cursor is positioned in a page description (for example between REPORT_PAGE and REPORT_PAGE_END).
- Bitmaps can only be **cut out**, if the cursor is located in a line requiring a bitmap indication, for example ICON, " ", (, , ,).

Positioning and page preview:

Mouse and cursor keys	Move the crosshair
<Left mouse button> or <ENTER>	Positioning or stop page preview
<Right mouse button> or <SPACE>	Specify the origin of the coordinate system

During element positioning, all elements already positioned are drawn in the **order** in which they are specified in the page description. This order is also valid for representation on the PCS 9000/PCS 9100. Furthermore, static elements are always located under dynamic elements (variables).

Display BMP:

Mouse or cursor keys	Move the crosshair
<Left mouse button> or <ENTER>	Stop BMP display
<Right mouse button> or <SPACE>	Specify the origin of the coordinate system

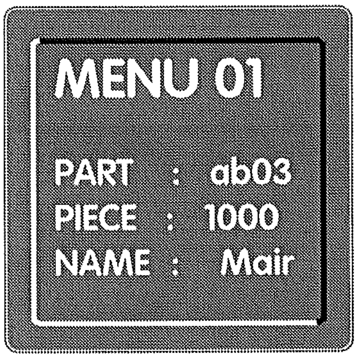
Cut out BMP

Mouse or cursor keys	Move the crosshair
<Left mouse button> or <ENTER>	Specify window or stop cut out
<Right mouse button> or <SPACE>	Switch between the following modes: - Enlarge/Reduce and - Move the rectangle

A bitmap range, including limitation given by the rectangle, is cut out.

Preset value entry using a clear menu structure 21

Like the operation, the number and the format of preset values are not uniform. The input procedure must always be clear and easy to understand, independent of the number of required preset values and their type.



MENU 01

PART : ab03
PIECE : 1000
NAME : Mair

Preset value input in a menu

The menu technique which offers a high flexibility is used to specify and change preset values, guide the operator and prevent wrong inputs as far as possible.

In the PCS

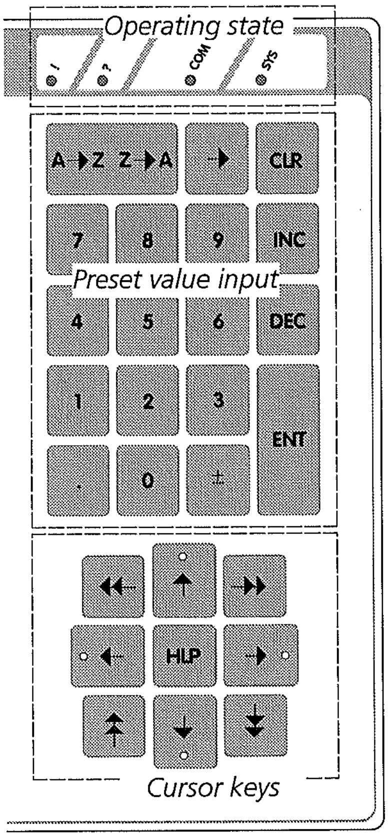
- 255 menus with a maximum of 64 variables in each of them

are available. The programmable controller calls a menu with word W 17 (bit 0...7). The LEDs in the arrow keys indicate to the operator in which direction other variables (preset values) can be edited, i.e. the corresponding LED is constantly lit. To close a menu, bit 0...7 in word 17 is reset.

For function and preset value input, a multifunctional editor is available in the PCS. This editor allows numbers to be entered in three different ways:

- Preset value input via the numeric keyboard
- Preset value correction via the INC/DEC keys
- Add or subtract any numeric values to/from the displayed preset value (only with BCD and BIN variables)

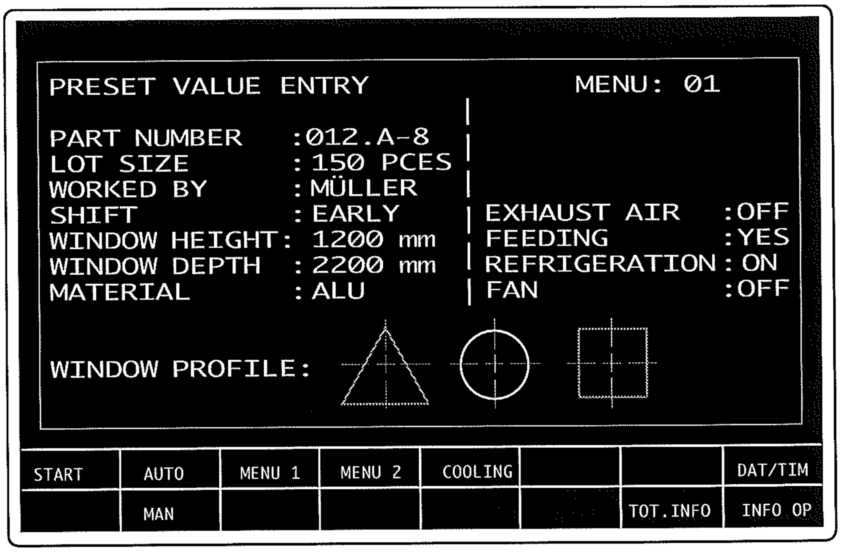
The CLR key is used to reset a preset value to the previous value.



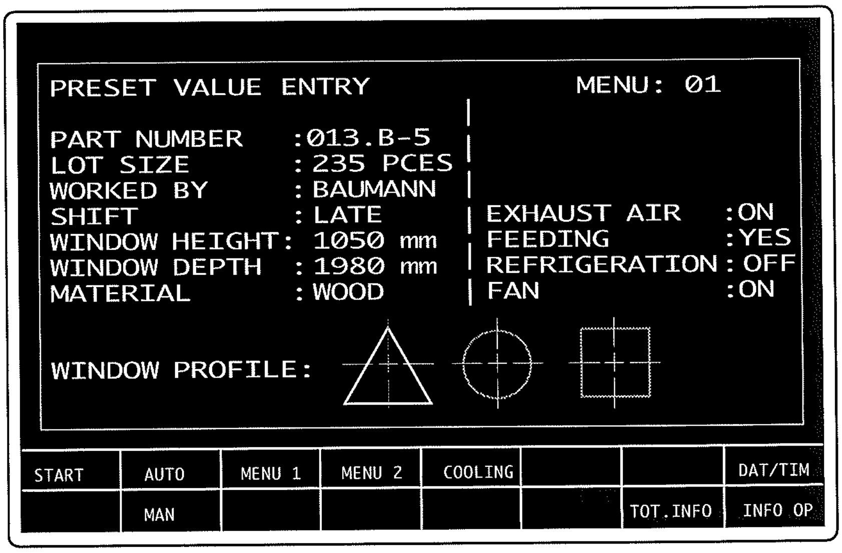
21 Preset value entry using a clear menu structure

PCS 9000 / PCS 9100

Examples for realizing different menus on the PCS*maxi* display. Combining graphic elements and texts in each of the four windows which are displayed simultaneously (Multi-Tasking Window System, MWS) is possible. Text and graphics variables can also be used together.

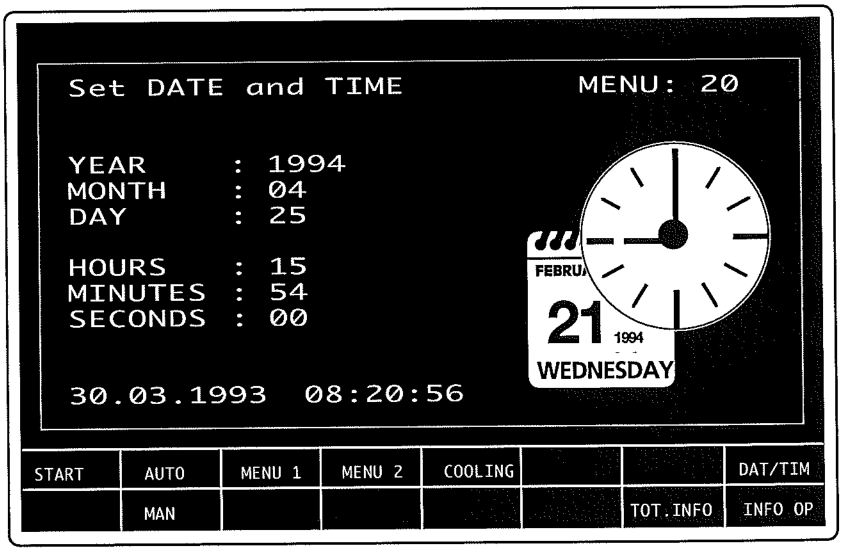


```
PRESET VALUE ENTRY                  MENU: 01

PART NUMBER     :012.A-8
LOT SIZE        :150 PCES
WORKED BY       :MÜLLER
SHIFT           :EARLY      EXHAUST AIR    :OFF
WINDOW HEIGHT:  1200 mm     FEEDING        :YES
WINDOW DEPTH    :2200 mm    REFRIGERATION  :ON
MATERIAL        :ALU        FAN            :OFF

WINDOW PROFILE:
```



```
PRESET VALUE ENTRY                  MENU: 01

PART NUMBER     :013.B-5
LOT SIZE        :235 PCES
WORKED BY       :BAUMANN
SHIFT           :LATE       EXHAUST AIR    :ON
WINDOW HEIGHT:  1050 mm     FEEDING        :YES
WINDOW DEPTH    :1980 mm    REFRIGERATION  :OFF
MATERIAL        :WOOD       FAN            :ON

WINDOW PROFILE:
```

Another menu example (setting clock and date). Apart from text elements, the menu also displays a static bit variable (clock with calendar).



```
Set DATE and TIME                   MENU: 20


YEAR      :  1994
MONTH     :  04
DAY       :  25

HOURS     :  15
MINUTES   :  54
SECONDS   :  00


30.03.1993   08:20:56
```

Steps for creation of an operating project

1 Determine the height of the soft-key labeling (50 pixels are sufficient for a two-line description per function key), the height of the status page, the message appearance (font and thus the heading window and message window height, the number of messages displayed simultaneously) and in this way the range covered by active messages. This determines the visible range on the idle page, if no menu, recipe or help are displayed.

2 The visible range on the idle page given by the height of the soft-key labeling and the status page can now be used to display company logos (images stored as BMP files), sequential function charts, machine images, statistics etc. Idle pages should be used to display information which are not edited.

3 Create (for example with Paintbrush), scan or import complex drawings, sequential function charts and process diagrams and convert them into the BMP format. Observe the memory capacity for static bitmaps. A complete image with 640x480 pixels requires 38400 Bytes.

4 Create graphic variables by cutting out the variables from the imported or created drawings, sequential function charts or process diagrams with Paintbrush. Then store the variables in a separate file and modify the different instances with Paintbrush.

5 Define all variables and assign the access class (actual and preset value) as well as the word or bit number.

6 Create idle pages, soft-key labelings, the status page, messages (with the priorities info, warning, fault and the different clear behaviors), operating pages, menus and help pages by inserting static and dynamic elements with the syntax help (function key <F1>) and by positioning them with the positioning help (function key <F6>).

7 Use the PCS 9092 software to compile the data record created in the PC or in the programmer under MSDOS/DRDOS or compatible DOS systems and transfer the compiled data record into the PCS.

8 Implement the handling software (PCS 91.nn, see appendix B) which is specific for the programmable controller into the user program and adjust the parameters.

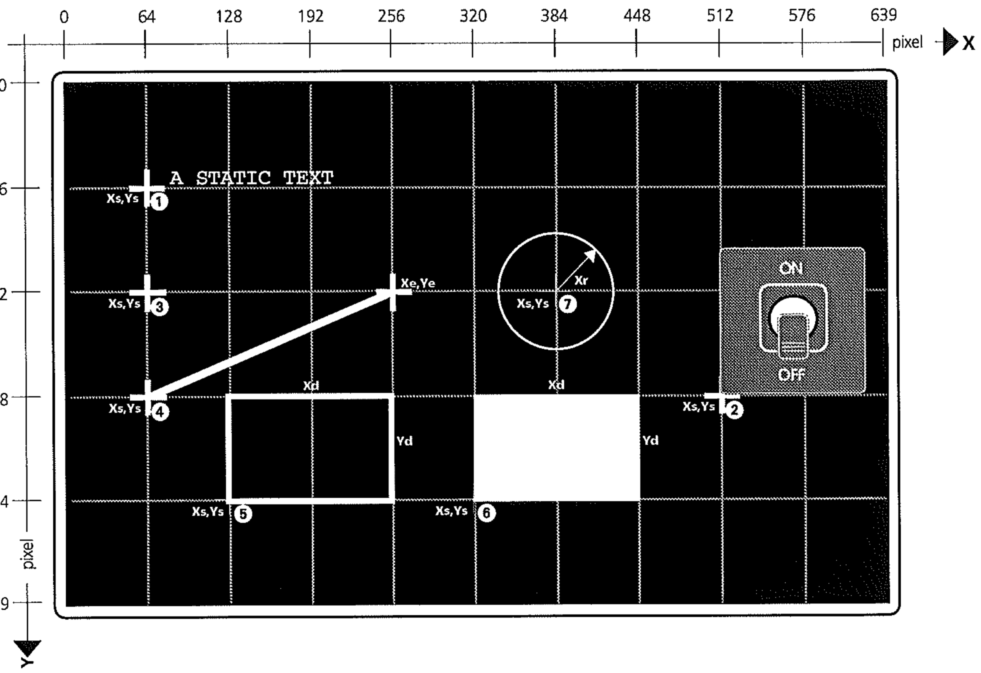
9 Connect the PCS and the programmable controller via the adapter cable. Test operation and control with the PCS and the programmable controller connected. Optimize the configuration, if required.

23 Positionable elements

	Idle page	Soft-key labeling	Status page	Messages		operat. page for menu	Help page	Operating printer page	Positio-ning	BMP-section	Fill pattern (16 x 16 pixels)	
				Heading page	mess. page (max. 8 words)						internal	BMP section
Static elements												
INS_ST_BITMAP	•	•	•		•	•	•		②	⑤		
INS_ST_TEXT	•	•	•	•	•	•	•	•	①			
INS_CIRCLE	•	•	•		•	•	•		⑦			
INS_PIXEL	•	•	•		•	•	•		③			
INS_LINE	•	•	•		•	•	•		④		1-12	⑤
INS_RECT	•	•	•		•	•	•		⑤			
INS_FILLRECT	•	•	•		•	•	•		⑥		1-12	⑤
Dynamic elements *external variable*												
BIT	•	•	•		•	•	•	•	①			
STRING	•	•	•		•	•	•	•	①			
CSTRING	•	•	•		•	•	•	•	①			
BCD-1	•	•	•		•	•	•	•	①			
BCD-2	•	•	•		•	•	•	•	①			
BCD0-1	•	•	•		•	•	•	•	①			
BCD0-2	•	•	•		•	•	•	•	①			
BIN-A	•	•	•		•	•	•	•	①			
BIN-B	•	•	•		•	•	•	•	①			
BIN0-A	•	•	•		•	•	•	•	①			
BIN0-B	•	•	•		•	•	•	•	①			
BIN-1	•	•	•		•	•	•	•	①			
BIN-2	•	•	•		•	•	•	•	①			
BIN0-1	•	•	•		•	•	•	•	①			
BIN0-2	•	•	•		•	•	•	•	①			
VBIN-A	•	•	•		•	•	•	•	①			
VBIN-B	•	•	•		•	•	•	•	①			
VBIN0-A	•	•	•		•	•	•	•	①			
VBIN0-B	•	•	•		•	•	•	•	①			
VBIN-1	•	•	•		•	•	•	•	①			
VBIN-2	•	•	•		•	•	•	•	①			
VBIN0-1	•	•	•		•	•	•	•	①			
VBIN0-2	•	•	•		•	•	•	•	①			
WORD	•	•	•		•	•	•	•	①			
ASCII	•	•	•		•	•	•	•	①			
GBIT	•	•	•		•	•	•		②	⑤		
GSTRING	•	•	•		•	•	•		②	⑤		
CGSTRING	•	•	•		•	•	•		②	⑤		
PHBAR	•	•	•		•	•	•		②		1-12	⑤
NHBAR	•	•	•		•	•	•		②		1-12	⑤
PVBAR	•	•	•		•	•	•		②		1-12	⑤
NVBAR	•	•	•		•	•	•		②		1-12	⑤
SHSYMBAR	•	•	•		•	•	•		②		1-12	⑤
SVSYMBAR	•	•	•		•	•	•		②		1-12	⑤
Dynamic elements *internal variable*												
[COM_ERRORS]	•	•	•	•	•	•	•	•	①			

Position specifications ①...⑤ please refer to the BMP coordinate system. The listing of the internal variables is for example purpose only and represents all internal variables

BMP coordinate system



Variable designators:

x_s, y_s ... position bottom left (for rectangles)
... start of line (INS_LINE)
... center of a circle (INS_CIRCLE)

x_e, y_e ... end of line (INS_LINE)

x_d, y_d ... size: width, height

x_r ... radius of a circle (INS_CIRCLE)

24 Data record example for an operating project

Complete data record of an operating project	Frame for global definitons, variables and page descriptions	Section	Entry	Description
Complete data record of an operating project	Info		DESCRIPTION	General project description (1024 characters max..)
			DESCRIPTIONEND	
			PCS9000_PARAMETER	
	Frame for global definitons, variables and page descriptions		BEGIN	Begin of the data record description
		Global definitions	LANGUAGEMAX, (1)	Number of languages (1..8) for the texts
			CHARSET (1), 852, 852	Specification of the character set for the first language: (1..9) or file name
			MSG_WINDOW, FONT (2), (20), 4	Font (1..8) for heading, message page height and number of messages
			MSG_DW_RANGE, 25, 10	Data word beginning for messages, number of message data words
			STATUS_WINDOW, (30)	Status page height in pixels (y_s=0..479)
			SKEYHIGH, (50)	Height of the horizontal soft-key labeling in pixels (yd=0..479)
			DATE_SIZE, EU	Date format (EU - DD.MM.YY, US - MM/DD/YY, MIL YY-MM-DD)
		Variable definitions – Text variable	VAR, Control voltage , 100, 1, BIT, 3	BIT variables definitions
			STATE_TEXT, 0	Instance 0
			LANGUAGE (1), "OFF"	Text language 1 for instance 0
			STATE_TEXT, 1	Instance 1
			LANGUAGE (1), "ON"	Text language 1 for instance 1
		Variable definitions – Graphic variable	VAR, Pneumatics, 101, 10, GBIT, 0	GBIT variables definitions
			STATE_ICON, 0	Instance 0
			ICON, "bild0.bmp", (264, 150, 24, 24)	Rectangular section of BMP file for instance 0
			STATE_ICON, 1	Instance 1
			ICON, "bild1.bmp", (304, 149, 24, 24)	Rectangular section of BMP file for instance 1
		Page descriptions – Idle page	REPORT_PAGE, 0	Description of idle page 0 (idle pages 0...255)
			INS_ST_BITMAP, (0, 399)	Static BITMAP, insert position (x_s, y_s)
			ICON, "LOGO.BMP", (0, 399, 640, 400)	Rectangular window (x_s, y_s, x_d, y_d) from BMP file
			REPORT_PAGE_END	End of description of idle page 0
		Soft-key labeling	SOFTKEY_ROW, 0	Page description of soft-key labeling 0 (soft-key labelings 0..255)
			INS_RECT, (0, 49), (640, 49)	Insert rectangular frame with position (x_s, y_s) and size (x_d, y_d)
			INS_ST_TEXT, (20, 18), FONT (1)	Labeling of function key 1, font specification (1...8)
			LANGUAGE (1), "Start"	Text for language 1 (LANGUAGE (2...), if multilingual)
			INS_ST_TEXT, (104, 18), FONT (1)	Labeling of function key 1, font specification (1...8)
			LANGUAGE (1), "Stop"	Text for language 1 (LANGUAGE (2...), if multilingual)
			INS_ST_TEXT, (165, 12), FONT (1)	Labeling of function key 1, font specification (1...8)
			LANGUAGE (1), "Contr. volt."	Text for language 1 (LANGUAGE (2...), if multilingual)
			INS_VAR, Control voltage, (189, 23), FONT (1)	Insertion of an external variable, position (x_s, y_s), font (1..8)
			SOFTKEY_ROW_END	End of page description of soft-key labeling 0
		Status	STATUS_PAGE	Description of the status page (located above in the display)
			INS_RECT, (0, 28), (640, 29)	Insert rectangular frame with position (x_s, y_s) and size (x_d, y_d)
			INS_ST_TEXT, (85, 24), FONT (2)	Insert static text, position (x_s, y_s), font specification (1...8)
			LANGUAGE (1), "Schnittstellenfehler:"	Text for language 1 (LANGUAGE (2...), if multilingual)
			INS_INTVAR, [COM_ERRORS], (442, 24), FONT (2)	Insertion of an internal variable, position (x_s, y_s), font (1..8)
			STATUS_PAGE_END	End of status page description
		Heading page	MSG_PAGE, TITEL	Description of the heading page
			INS_ST_TEXT, (45, 19)	Insert static text, position (x_s, y_s), font specification (1...8)
			LANGUAGE (1), „Message came: : time"	Text for language 1 (LANGUAGE (2...), if multilingual)
			INS_INTVAR, [CLK_HOUR], (428, 19)	Insertion of an internal variable, position (x_s, y_s), font (1..8)
			MSG_PAGE_END	End of heading page description
		Message page	MSG_PAGE, 0 , INFO , QUIT	Description of message page 0 (0...1023)
			INS_ST_TEXT, (192, 19), FONT (2)	Insert static text, position (x_s, y_s), font specification (1...8)
			LANGUAGE (1), „Close doors!"	Text for language 1 (LANGUAGE (2...), if multilingual)
			MSG_PAGE_END	End of description of message page 0
		Operating page	OP_PAGE, 1 (400, 100)	Description of the operating page 1 (0...255 operating pages)
			INS_ST_TEXT, (88, 69), FONT (3)	Insert static text, position (x_s, y_s), font specification (1...8)
			LANGUAGE (1), "Pneumatik"	Text for language 1 (LANGUAGE (2...), if multilingual)
			INS_VAR, Pneumatik, (33, 62)	Insertion of an external graphics variable (defined above)
			OP_PAGE_END	End of description of operating page 1
		Menu	MENU, Pneumatics, 1	Define menu with name and number (1...255 menus)
			NODE, 1, (128, 150)	Assign operating page 1, position (x_s, y_s) on the display
		Help page	HLP_PAGE, 1, MENU, (128, 150), (400, 100)	Description of help page 1 with priority (here menu)
			INS_ST_TEXT, (51, 55), FONT (1)	Insert static text, position (x_s, y_s), font specification (1...8)
			LANGUAGE (1), "Toggle with INC, DEC"	Text for language 1 (LANGUAGE (2...), if multilingual)
			HLP_PAGE_END	End of description of help page 1
			INCLUDE "RP001.INC"	Insert page descriptions and variables definitions in other operating project files
			END	End of data record description

Example for planning a project in practice 25

The PCS 9092 software is equipped with an operator interface which corresponds to the SAA standard. Operation is possible via the mouse and/or the keyboard.

The example is divided into 9 steps. The result of step 1 is stored in the file *STEP\STEP01.PCS*. So you can compare the result of your entry with the requirements and also terminate programming and continue later by loading the corresponding data record.

The following example explains, step by step, the creation of an operating project, the use of the project planning software PCS 9092, the editor and the positioning tools of PCS*maxi*. In this project planning software, the following variables and pages are described:

1 Create projects and specify global project data

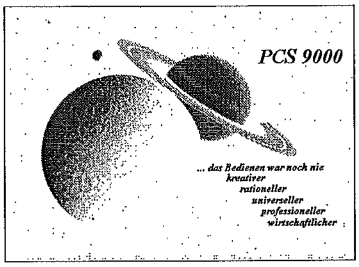


Image of the idle page (static BITMAP 640 x 480 pixels)

2 Specification of variables

- BIT variable control voltage in soft-key labeling 0
  with the instances OFF and ON
- GBIT variable pneumatics in menu 1
  with the instances ⏻ and ⦶ which is taken from the supplied BITMAP library LIB_MECH\MECH01.BMP.

3 Idle page

- Idle page 0,
  with an image from the file STEP\LOGO.BMP, resolution of 640x480 pixels

4 Soft-key labeling

- Soft-key labeling 0,
  with the following assignment:

	F-key F1	*START*
	F-key F2	*STOP*
	F-key F3	BIT variable *control voltage*

5 Status page

- Information concerning the external variable [COM_ERRORS]
  This variable gives information about the quality of the connection between the PCS and the programmable controller.

6 Messages

• Message heading for all messages	*Message, time*
• message 0 as info	*Close doors*
• message 1 as warning	*Part feeder empty*
• message 2 as fault	*Overcurrent trip M1 triggered*

7 Operating page for menu

- Operating page 1
  with the graphic GBIT variable *Pneumatics*

8 Menu

- Menu 1, *Pneumatics*
  Operating page 1 is linked

9 Help page

- Help for operating page 1
- Help for message 0

25 Example for planning a project in practice

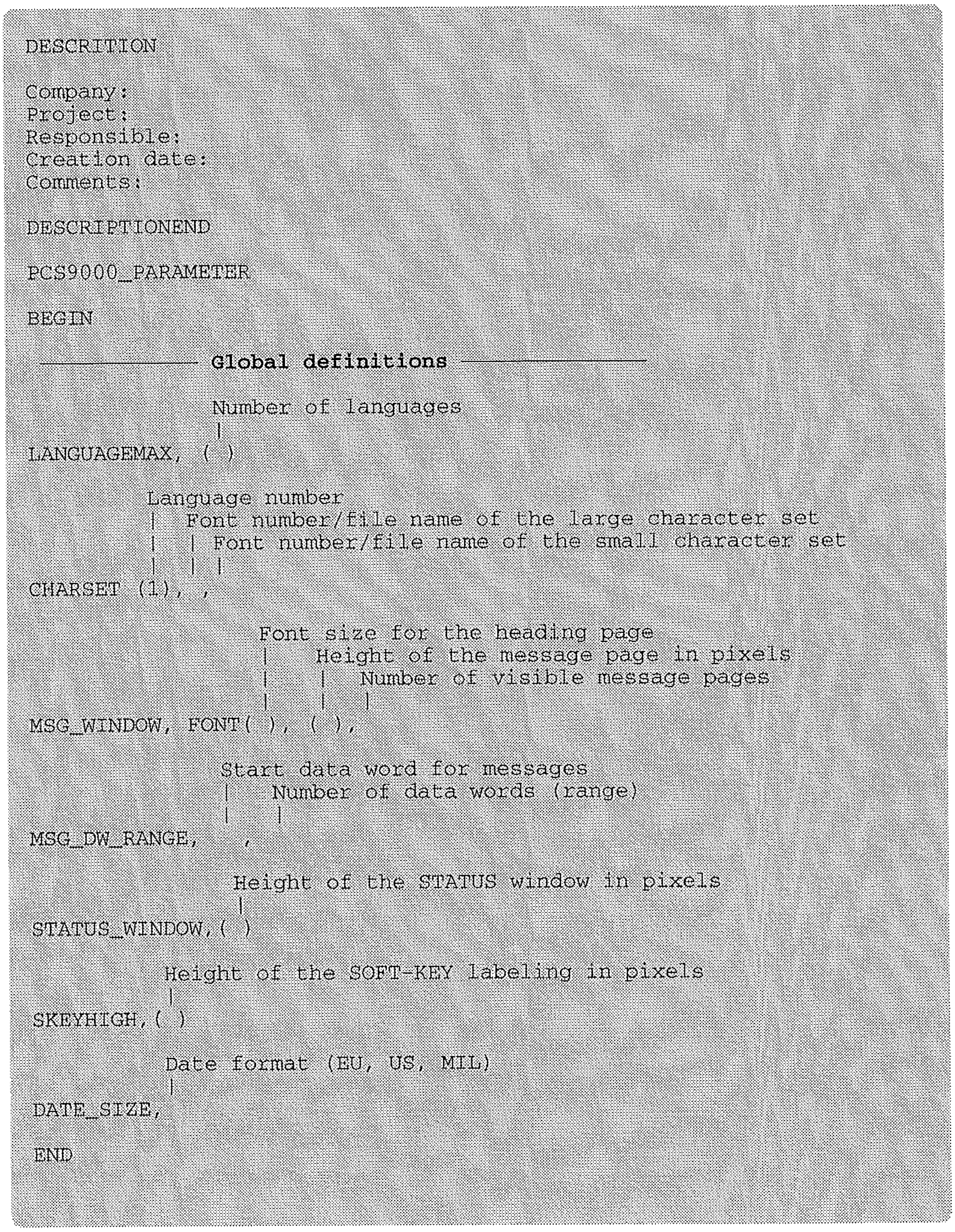
The following conventions are valid for the following descriptions:

Position the cursor	⇗
Mouse action	🖱
Keybord entry	⌨

Step 1

Create a new operating project and specify global project data

- Execute PCS 9092 with P.BAT <ENTER>
- <Alt-D>
  „New" <ENTER>
- <F1>
  „Project body" <ENTER>



```
DESCRITION

Company:
Project:
Responsible:
Creation date:
Comments:

DESCRIPTIONEND

PCS9000_PARAMETER

BEGIN

 ———————————— Global definitions ————————————

              Number of languages
              |
LANGUAGEMAX, ( )

         Language number
         |  Font number/file name of the large character set
         |  | Font number/file name of the small character set
         |  | |
CHARSET (1), ,

                  Font size for the heading page
                  |   Height of the message page in pixels
                  |   |  Number of visible message pages
                  |   |  |
MSG_WINDOW, FONT( ), ( ),

               Start data word for messages
               |   Number of data words (range)
               |   |
MSG_DW_RANGE,    ,

                Height of the STATUS window in pixels
                |
STATUS_WINDOW,( )

          Height of the SOFT-KEY labeling in pixels
          |
SKEYHIGH,( )

          Date format (EU, US, MIL)
          |
DATE_SIZE,

END
```

- Input of the project data

```
DESCRITION

Company:          Systeme LAUER GmbH
                  Kelterstrasse 59
Project:          Example STEP01.PCS
Responsible:      ak, SIC
Creation date:    25.01.1994
Comments:

DESCRIPTIONEND

PCS9000_PARAMETER

BEGIN

 ———————————— Global definitions ————————————

             One language
              |
LANGUAGEMAX, (1)

         Language number
         |  Font number/file name of the large character set
         |  | Font number/file name of the small character set
         |  | |
CHARSET (1), ,

                 Font size for the heading page
                 |    Height of the messsage page in pixels
                 |    |   4 visible message pages
                 |    |   |
MSG_WINDOW, FONT(2),(20),4

             Start data word for messages
             |  Number of data words for messages
             |  |
MSG_DW_RANGE,25,10

               Height of the STATUS window in pixels
               |
STATUS_WINDOW,(30)

          Height of the SOFT-KEY- labeling in pixels
          |
SKEYHIGH,(50)

          Date format EU (European)
          |
DATE_SIZE,EU

END
```

BOLD

User

entered

- Finally, check correct syntax with <F9>.

The data record created until now should correspond to the one in the file STEP\STEP01.PCS.

Summary of step 1

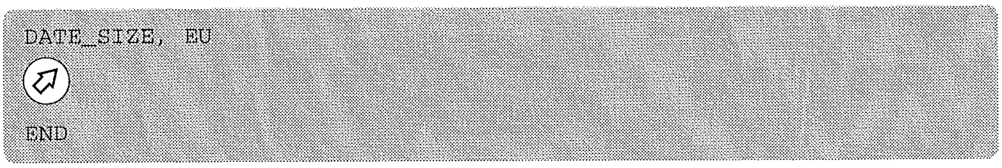
- *The syntax help (function key <F1>) was used to insert the syntax for the project body (project description with DESCRIPTION ... DESCRIPTIONEND) and the parameterization range (PCS9000_PARAMETER: BEGIN ... END). This range now contains global definitions.*
- *The global definitions are generally made at the beginning of the project planning. Here, the project engineer specifies the height of the soft-key labeling, the status page, the message window and the individual messages as well as the format of date and time and the number of languages. After the global definitions, internal variables (e.g. [LANGUAGE], [COM_ERRORS]...) are parameterized. Variables declarations and page descriptions are then inserted into this project base.*
- *Comments and commands are distinguished by leading spaces.*
  - *Commands may not contain any leading spaces and must be written in uppercase letters.*
  - *Comments have at least one leading space.*

Step 2

Specify variables

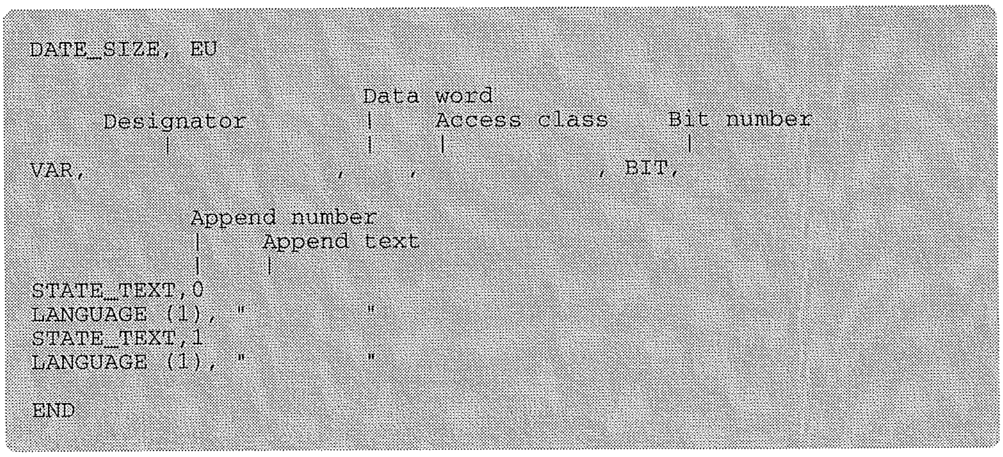
The following operations are based on the state specified in the file STEP\STEP01.PCS.

- Position the cursor in the line before the last instruction "END".



```
DATE_SIZE, EU

END
```

- <F1>
  „Variables" <ENTER>
  „BIT" <ENTER>



```
DATE_SIZE, EU

                       Data word
     Designator        |    Access class    Bit number
          |            |    |               |
VAR,                 ,    ,            , BIT,

           Append number
           |    Append text
           |    |
STATE_TEXT,0
LANGUAGE (1), "        "
STATE_TEXT,1
LANGUAGE (1), "        "

END
```

- please complete:

```
                         Data word
     Designator          |    Access class      Bit number
     |                   |    |                      |
VAR, Control voltage   , 100, 1          , BIT, 3
             AP number
             |
STATE_TEXT, 0
                  AP text
                  |
LANGUAGE (1), "OFF"
STATE_TEXT, 1
LANGUAGE (1), "ON"

END
```

- position the cursor :

```

LANGUAGE (1), "OFF"
STATE_TEXT, 1
LANGUAGE (1), "ON"
↗

END
```

- <F1>
  „Variables" <ENTER>
  „GBIT" <ENTER>
- and complete as follows

```
                         Data word
     Designator          |   Access class        Bit number
     |                   |   |                      |
VAR, Pneumatics        , 101, 1          , GBIT, 0
             AP number
             |
STATE_ICON, 0
        Bitmap file name          Xs   Ys   Xd   Yd  (rectangle
section of the icon)
             |                    |    |    |    |
ICON,"..\lib_mech\mech01.bmp", (264, 150,  24,  24)
STATE_ICON, 1
ICON,"..\lib_mech\mech01.bmp", (304, 149,  24,  24)

END
```

- Finally, check the syntax with <F9>.

The data record created until now should correspond to the one in the file *STEP\STEP02.PCS*.

Summary of step 2

In step 2, all variables used in this project were defined. Variables can be defined immediately before the description of the page where the variable is used.

Data from the BMP file can also be selected in an interactive manner by positioning the cursor in the ICON line and cutting out a section with

- *<F5> <ENTER>*
  "..\" <ENTER>
  "LIB-MECH" <ENTER>
  "MECH01.BMP" <ALT-T>

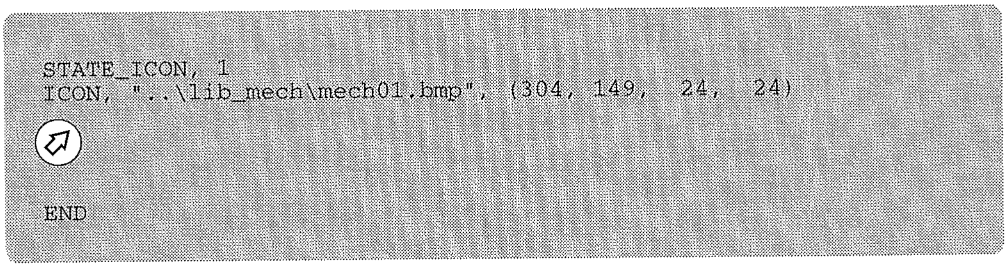
The <SPACE> bar or the <right mouse button> is used to switch between ENLARGE/ REDUCE and MOVE. <ENTER> or the <left mouse button> specify the window.

Step 3

Create an idle page and insert the static bitmap as background image

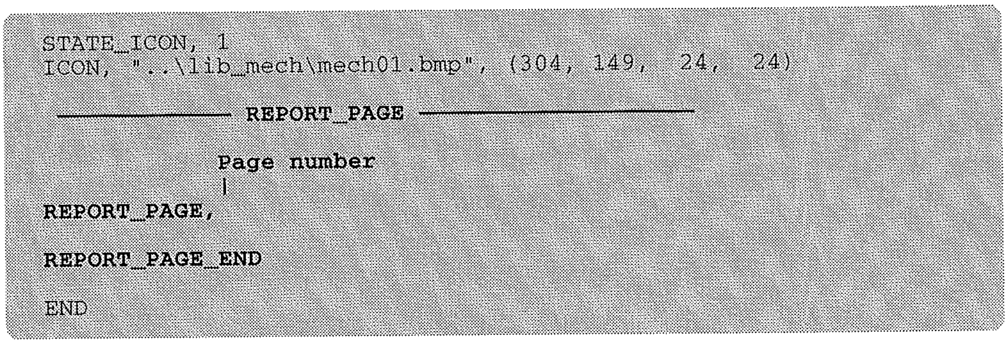
The following operations are based on the state specified in the file *STEP\STEP02.PCS.*

- Position the cursor in the line before the last instruction"END".



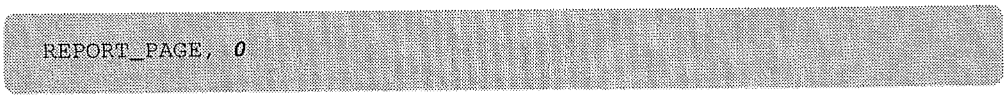
```
STATE_ICON, 1
ICON, "..\lib_mech\mech01.bmp", (304, 149,  24,  24)



END
```

- <F1>
  „Idle page" <ENTER>



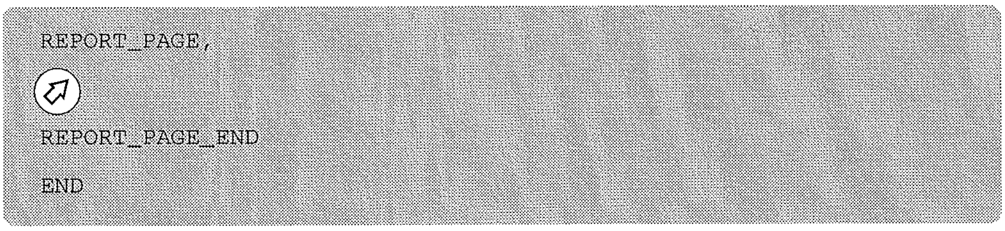
```
STATE_ICON, 1
ICON, "..\lib_mech\mech01.bmp", (304, 149,  24,  24)

  ———————————— REPORT_PAGE ————————————————————

                Page number
                |
REPORT_PAGE,

REPORT_PAGE_END

END
```

- Position the cursor in the following line and specify the number of the idle page:



```
REPORT_PAGE, 0
```

- Position the cursor in the line before REPORT_PAGE_END.



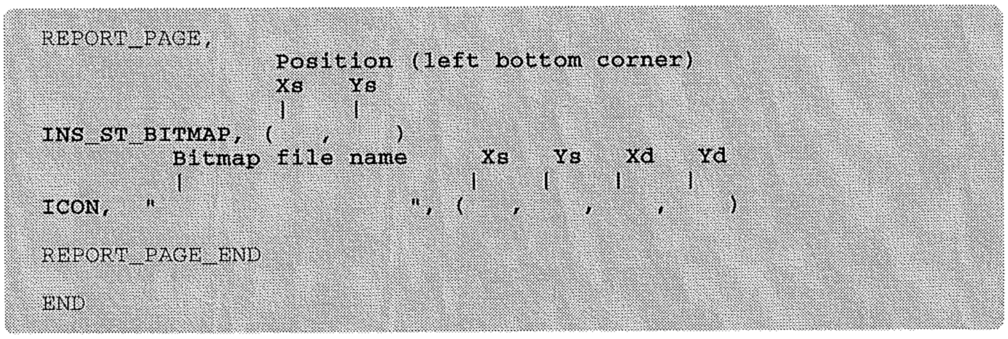
```
REPORT_PAGE,


REPORT_PAGE_END

END
```

Generally, the idle page size is 640x480 pixels.

After having specified the start (REPORT_PAGE) and the end (REPORT_PAGE_END) of the idle page description, the elements which have to be displayed on the PCS when retrieving this page can be inserted between start and end. You can insert all external and internal variables already defined and all static elements (bitmap, text, circle, line, point, rectangle and filled rectangle).

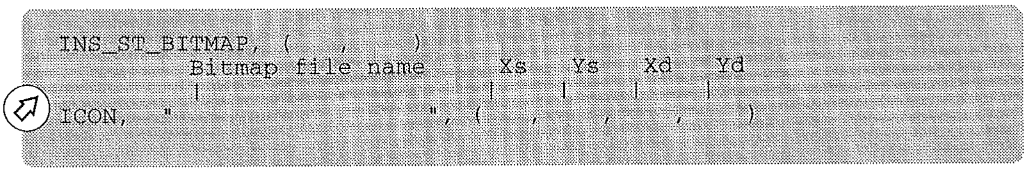
Now insert the image from the file STEP\LOGO.BMP as static bitmap (background image).

- <F1>
  „Insert element" <ENTER>
  „Static bitmap" <ENTER>



```
REPORT_PAGE,
                  Position (left bottom corner)
                  Xs   Ys
                  |    |
INS_ST_BITMAP, (    ,    )
         Bitmap file name     Xs   Ys   Xd   Yd
         |                    |    |    |    |
ICON,  "                  ", (   ,    ,    ,    )

REPORT_PAGE_END

END
```

- Position the cursor in the line „ICON" and



```
INS_ST_BITMAP, (    ,    )
         Bitmap file name     Xs   Ys   Xd   Yd
         |                    |    |    |    |
ICON,  "                  ", (   ,    ,    ,    )
```

- using <F5>
  <ENTER>
  „STEP\" <ENTER>
  „LOGO.BMP" <Cutout, all>
  cutout the complete contents of the file. Next, the following entry is shown on the ICON line

```
                Position (left bottom corner)
                Xs   Ys
                |    |
INS_ST_BITMAP, (    ,    )
        Bitmap file name       Xs    Ys    Xd    Yd
        |                      |     |     |     |
ICON, "LOGO.BMP",             (0, 399, 640, 400)
```

- Position the cursor in the line INS_ST_BITMAP, (,)

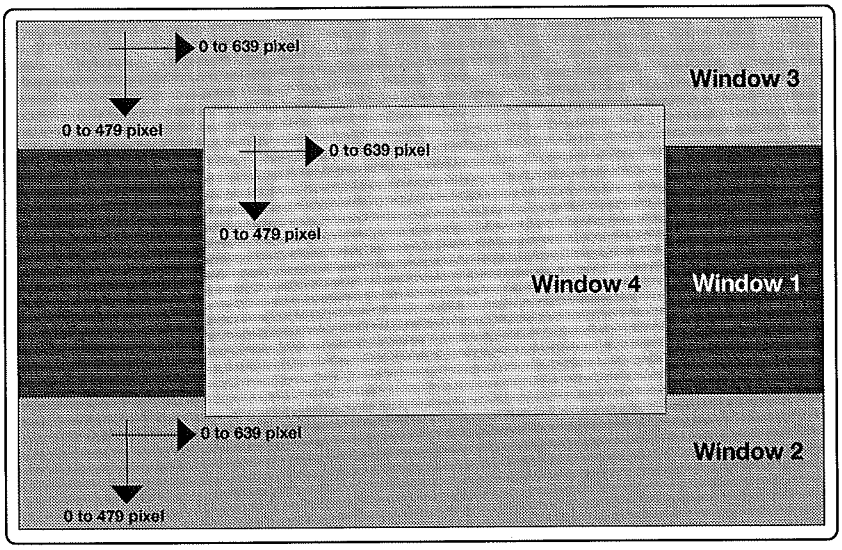
```
                Position (left bottom corner)
                Xs   Ys
                |    |
INS_ST_BITMAP, (    ,    )
        Bitmap file name       Xs    Ys    Xd    Yd
        |                      |     |     |     |
ICON, "LOGO.BMP",             (0, 399, 640, 400)
```

- and load the positioning help by pressing <F6>.
- Press the <left mouse button> or <ENTER> for bitmap positioning.
- Press the <left mouse button> or <ENTER> to terminate the positioning help.
- Finally, check correct syntax with <F9>.

The data record created until now should correspond to the one in the file STEP\STEP03.PCS.

Summary of step 3

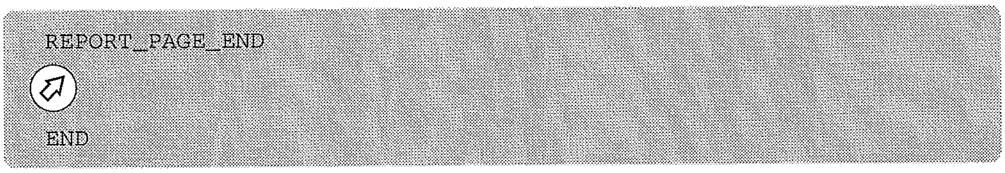
The idle page (window 1) is located in the background and generally has a size of 640x480 pixels. It is covered by all other windows (soft-key labeling, status page, messages, menus, recipes and help information). This has to be observed during project planning, if the idle page is for example used to represent a process. In this case, leave enough space on the idle page for the height of the soft-key labeling (lower margin) and the height of the status page and, if required, of the messages (upper margin).



Create a soft-key labeling and insert a static text and a variable **Step 4**

The following operations are based on the state specified in the file STEP\STEP03.PCS.

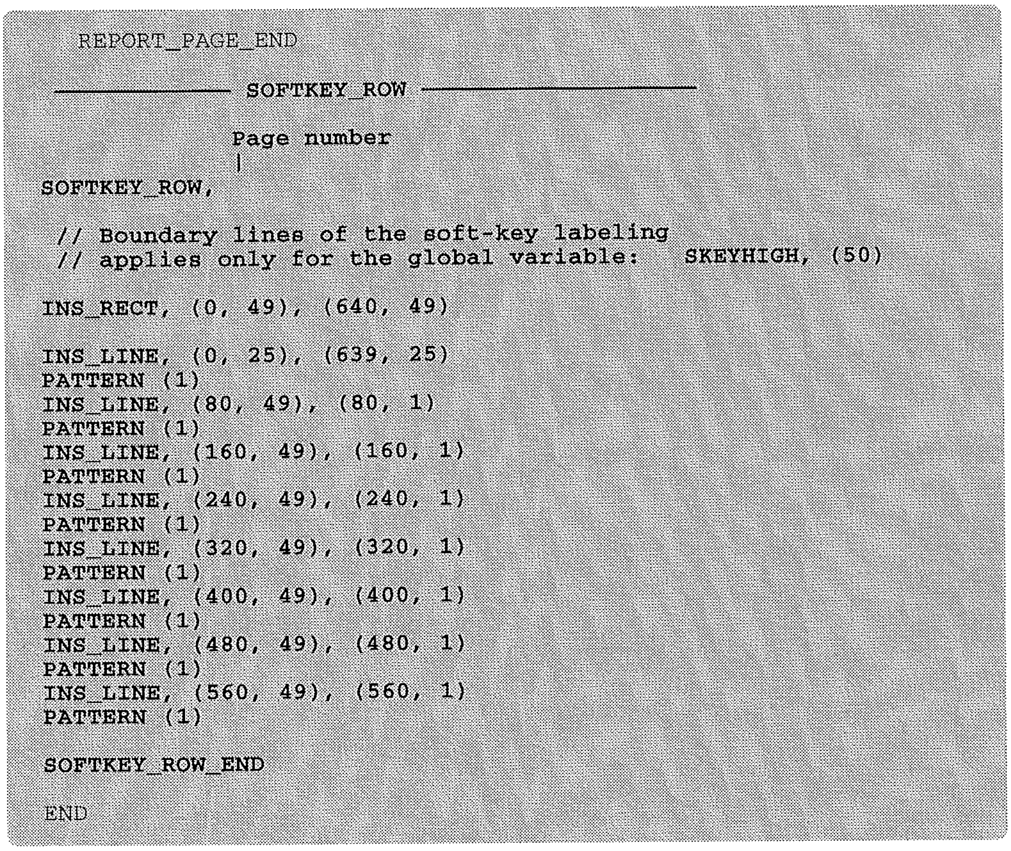
- Position the cursor in the line before the command "*END*".



```
REPORT_PAGE_END


END
```

Now insert the page description of a soft-key labeling with predefined grid for the 16 function keys. The different static elements (INS_RECT, INS_LINE) generate the 16 function key fields. The height of the generated soft-key labeling is 50 pixels.

- <F1>
  „Soft-key labeling"
  <TAB> <TAB> <SPACE> „with boudaries" <ENTER>



```
  REPORT_PAGE_END

 ───────────── SOFTKEY_ROW ─────────────────────

               Page number
               |
SOFTKEY_ROW,

 // Boundary lines of the soft-key labeling
 // applies only for the global variable:   SKEYHIGH, (50)

INS_RECT, (0, 49), (640, 49)

INS_LINE, (0, 25), (639, 25)
PATTERN (1)
INS_LINE, (80, 49), (80, 1)
PATTERN (1)
INS_LINE, (160, 49), (160, 1)
PATTERN (1)
INS_LINE, (240, 49), (240, 1)
PATTERN (1)
INS_LINE, (320, 49), (320, 1)
PATTERN (1)
INS_LINE, (400, 49), (400, 1)
PATTERN (1)
INS_LINE, (480, 49), (480, 1)
PATTERN (1)
INS_LINE, (560, 49), (560, 1)
PATTERN (1)

SOFTKEY_ROW_END

END
```

- Specify the soft-key labeling number.

```
SOFTKEY_ROW, 0
```

- Position the cursor in the line above SOFTKEY_ROW_END.

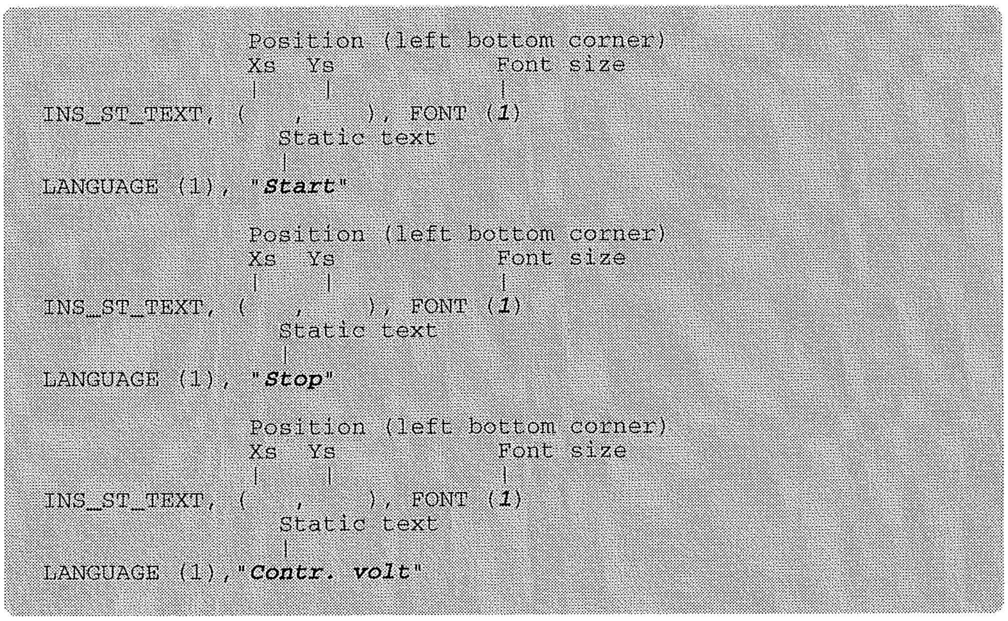
```
INS_LINE, (560, 49), (560, 1)
PATTERN (1)
➚

SOFTKEY_ROW_END

END
```

Three static texts and the variable control voltage are then inserted. For this purpose, repeat the following entry three times.

- <F1>
  „Insert element " <ENTER>
  „Static text" <ENTER>

```
INS_LINE, (560, 49), (560, 1)
PATTERN (1)

               Position (left bottom corner)
               Xs  Ys             Font size
               |     |            |
INS_ST_TEXT, (   ,    ), FONT ( )
                 Static text
                 |
LANGUAGE (1), "                   "

               Position (left bottom corner)
               Xs  Ys             Font size
               |     |            |
INS_ST_TEXT, (   ,    ), FONT ( )
                 Static text
                 |
LANGUAGE (1), "                   "

               Position (left bottom corner)
               Xs  Ys             Font size
               |     |            |
INS_ST_TEXT, (   ,    ), FONT ( )
                 Static text
                 |
LANGUAGE (1), "                   "

SOFTKEY_ROW_END

END
```

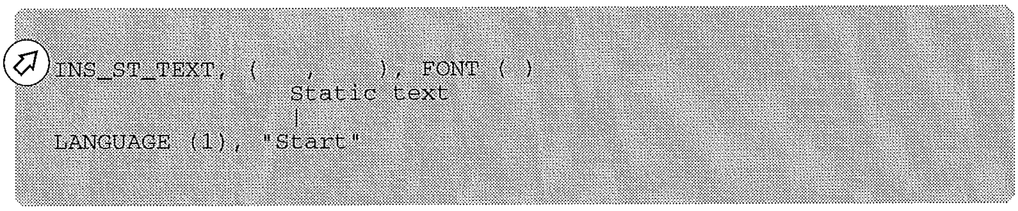
- Complete the 3 static texts



```
                 Position (left bottom corner)
                 Xs   Ys             Font size
                 |    |              |
INS_ST_TEXT, (     ,     ), FONT (1)
                   Static text
                   |
LANGUAGE (1), "Start"

                 Position (left bottom corner)
                 Xs   Ys             Font size
                 |    |              |
INS_ST_TEXT, (     ,     ), FONT (1)
                   Static text
                   |
LANGUAGE (1), "Stop"

                 Position (left bottom corner)
                 Xs   Ys             Font size
                 |    |              |
INS_ST_TEXT, (     ,     ), FONT (1)
                   Static text
                   |
LANGUAGE (1),"Contr. volt"
```

Now, position the three texts. Move the cursor successively into the three INS_ST_TEXT lines and load the positioning help by pressing <F6>:

- Position the cursor:

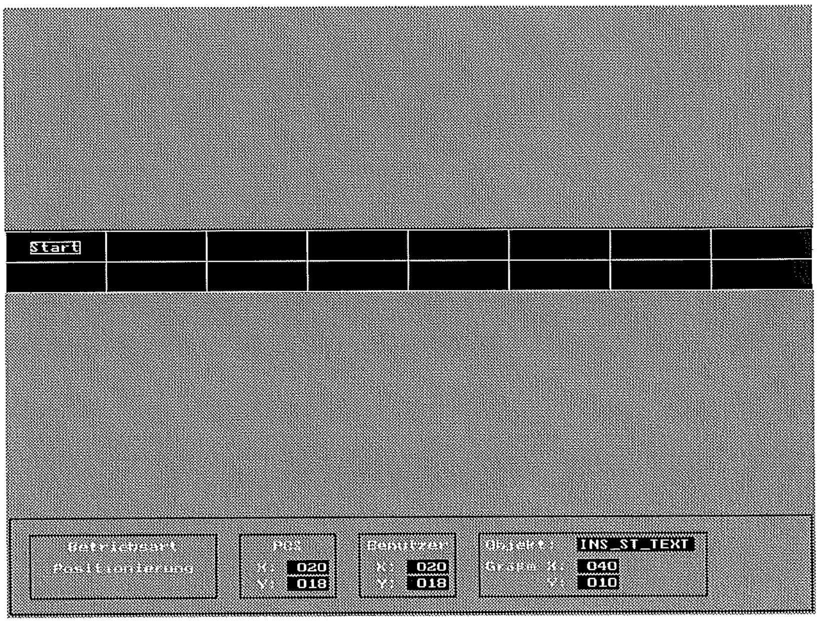


```
INS_ST_TEXT, (     ,     ), FONT ( )
                   Static text
                   |
LANGUAGE (1), "Start"
```

- <F6>

Use the mouse or the cursor keys for centering the rectangle in the first field. Then position the text by pressing the <left mouse button> or <ENTER>. The following is displayed:

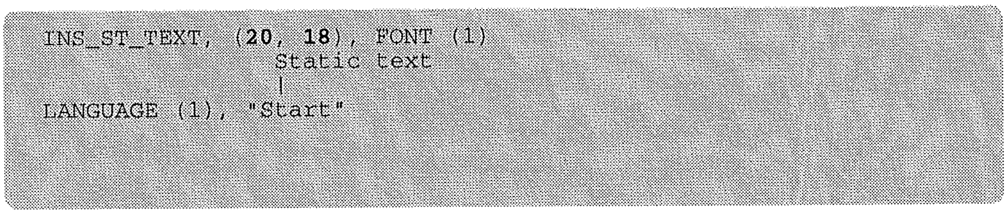
- <left mouse button> or <ENTER>



```
Start

Betriebsart      PCS         Benutzer     Objekt:  INS_ST_TEXT
Positionierung   X:  020     X:  020      Größe X: 040
                 Y:  018     Y:  018            Y: 010
```

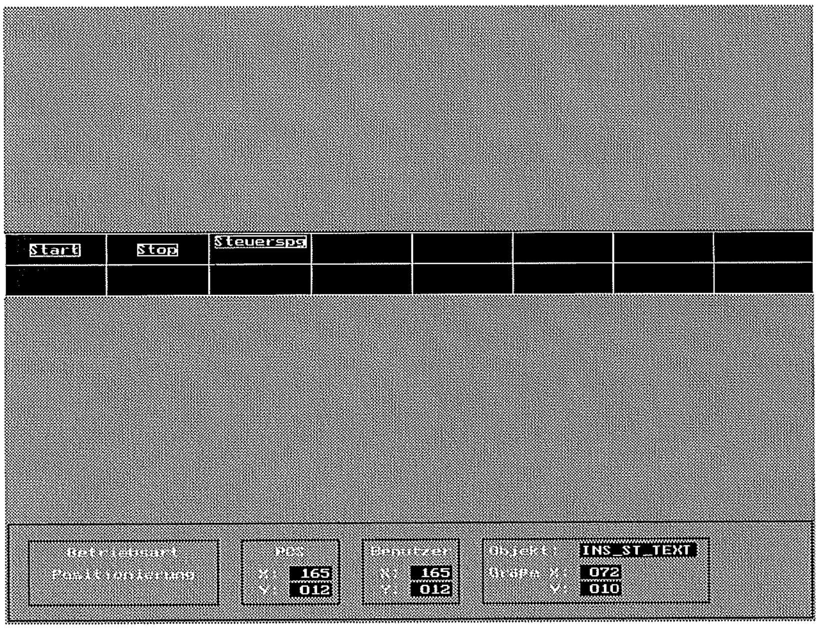
Press the <left mouse button> or <ENTER> again to terminate the positioning help and to insert the coordinates in the editor as follows:

- <left mouse button> or <ENTER>



```
INS_ST_TEXT, (20, 18), FONT (1)
                Static text
                   |
LANGUAGE (1), "Start"
```

Position now the next two static texts, as represented below: Move the cursor on INS_ST_TEXT and press <F6> for positioning.



Betriebsart
Positionierung

PCS
X: 165
Y: 012

Benutzer
X: 165
Y: 012

Objekt: INS_ST_TEXT
Größe X: 072
Y: 010

The last element on this page (variable Control Voltage) is positioned as follows:

- Position the cursor:

```
LANGUAGE (1), "Contr. volt"

SOFTKEY_ROW_END
END
```

Insert the variable *Control voltage*:

- <F1>
  „Insert element" <ENTER>
  „Variables" <ENTER>

```
                   Static text
                   |
LANGUAGE (1), "                  "

                                   Position (left bottom corner)
         Designator                  Xs    Ys        Font size
         |                           |     |             |
INS_VAR,                           , (     ,     ),  FONT ( )

SOFTKEY_ROW_END
```

- please complete:

```
                          Position (left bottom corner)
         Designator       Xs   Ys          Font size
         |                |    |           |
INS_VAR, Control voltage  , (    ,    ),  FONT (1)
```

Finally, position the cursor below the text „Contr. volt" using the key <F6>

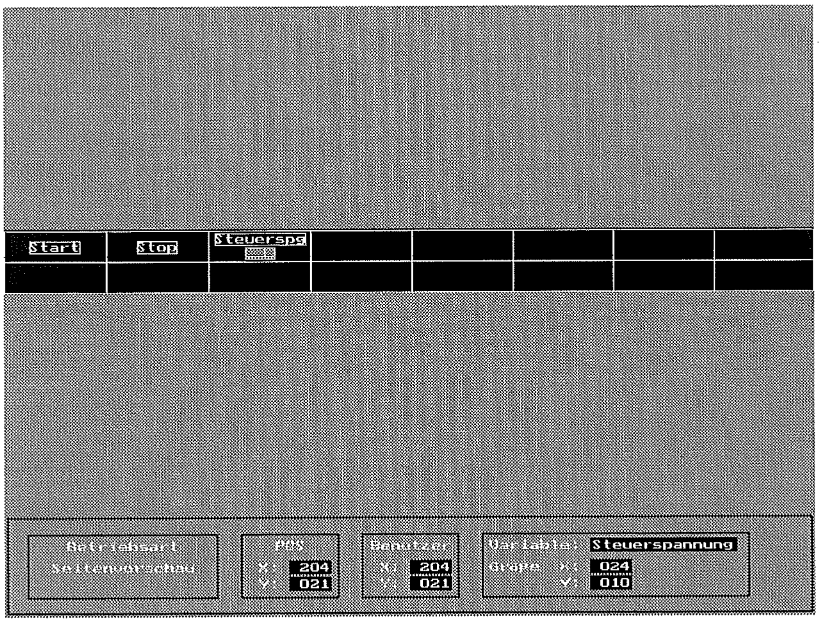
- <F6>
  <left mouse button> or <ENTER>
  <left mouse button> or <ENTER>

In the editor you should get similar positioning data as follows:

```
INS_LINE, (560, 49), (560, 1)
PATTERN (1)

              Position (left bottom corner)
              Xs  Ys           Font size
              |    |           |
INS_ST_TEXT, (20, 18), FONT (1)
                Static text
                |
LANGUAGE (1), "Start"

              Position (left bottom corner)
              Xs  Ys           Font size
              |    |           |
INS_ST_TEXT, (104, 18), FONT (1)
                Static text
                |
LANGUAGE (1), "Stop"

              Position (left bottom corner)
              Xs  Ys           Font size
              |    |           |
INS_ST_TEXT, (165, 12), FONT (1)
                Static text
                |
LANGUAGE (1), "Contr. volt"

                         Position (left bottom corner)
         Designator      Xs   Ys          Font size
         |               |    |           |
INS_VAR, Control voltage, (189, 23), FONT (1)

SOFTKEY_ROW_END
```

All elements of the soft-key labeling are now defined. The positions were specified with the positioning help (<F6> key). If all elements are positioned, press <Ctrl-F6> (page preview) to display the complete page with all positioned elements.

- <Ctrl-F6>



If the crosshair is positioned on a variable during page preview, the designator and the variable size are displayed in the lower right corner. To terminate the page preview, press the <left mouse button> or <ENTER>.

- Finally, check correct syntax with <F9>.

The data record created until now should correspond to that in the file STEP\STEP04.PCS.

Summary of step 4

- *Generally, the soft-key labeling has a size of 640 pixels and is located at the lower margin of the display. The height can be set as required between 0 and 480. A visible soft-key labeling always covers the idle page (REPORT_PAGE) which is located in the background.*
- *In step 4, the syntax help (<F1> key) was used to insert the page description and to specify the soft-key labeling (16 function keys were assigned).*
- *Different elements - a static text and a variable - were then inserted with the syntax help.*
- *The positioning help (<F6> key) was used to determine the position of the individual elements.*
- *The page preview <Ctrl-F6> was used to display all elements inserted in a page description between the beginning (e.g. REPORT_PAGE, SOFTKEY_ROW etc.) and the end (e.g. REPORT_PAGE_END, SOFTKEY_ROW_END etc.). A gray rectangle is displayed for the inserted variables. If the crosshair is moved on a variable, the designator and the size are displayed in the information area.*

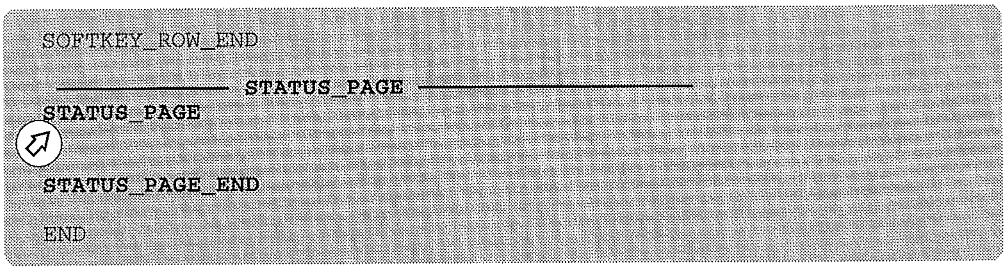
Step 5

Create a status page

The following operations are based on the state specified in the file STEP\STEP04.PCS.

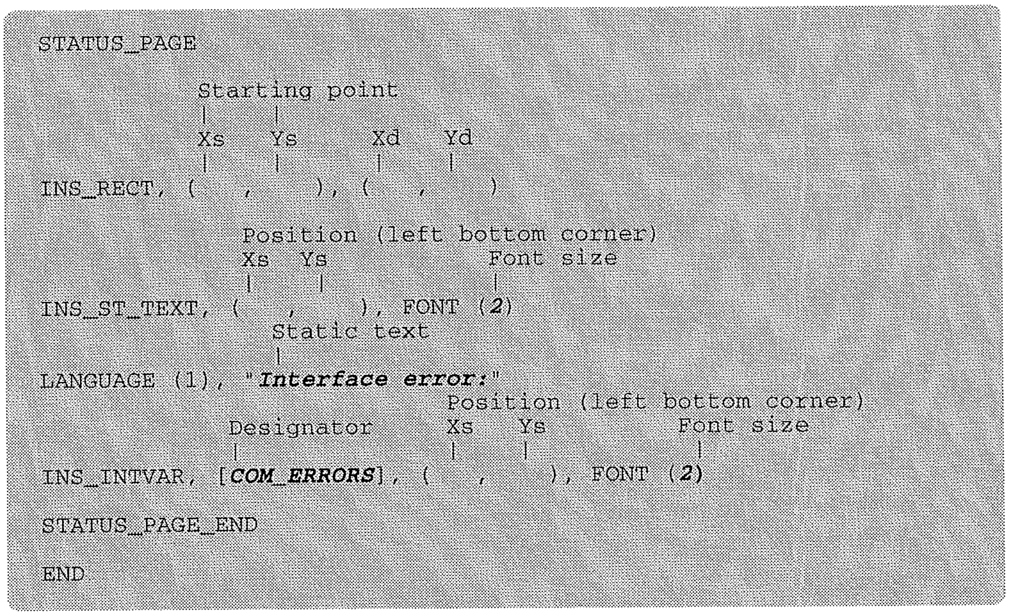
On the status page, information such as date, time, machine operating conditions, interface errors etc. are generally displayed.

Using the syntax help for other page elements corresponds to steps 1 to 4. Please insert now the description of a status page after the soft-key labeling description and before the last command END. For this purpose, load the syntax help and position the cursor on the page:

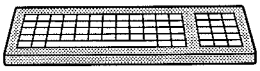


```
SOFTKEY_ROW_END

————————————— STATUS_PAGE ——————————————————
STATUS_PAGE

STATUS_PAGE_END

END
```

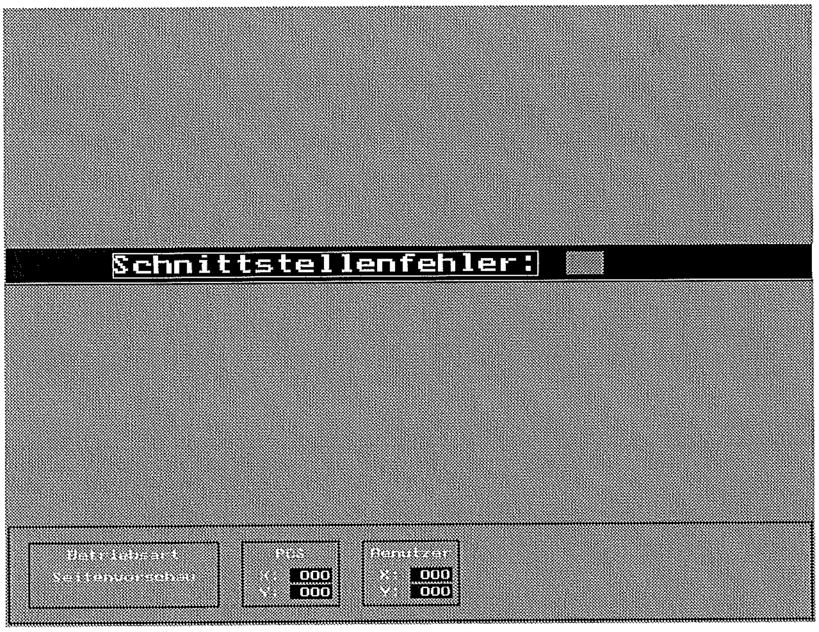
Then use the syntax help to insert a window frame with INS_RECT, a static text and the internal variable [COM_ERRORS]. The internal variable informs the operator about the interface error occurred in the connection between the PCS and the programmable controller.



```
STATUS_PAGE

           Starting point
           |    |
           Xs   Ys     Xd   Yd
           |    |      |    |
INS_RECT, (   ,    ), (   ,    )

              Position (left bottom corner)
              Xs  Ys           Font size
              |   |            |
INS_ST_TEXT, (   ,    ), FONT (2)
                Static text
                |
LANGUAGE (1), "Interface error:"
                            Position (left bottom corner)
             Designator     Xs   Ys          Font size
             |              |    |           |
INS_INTVAR, [COM_ERRORS], (   ,    ), FONT (2)

STATUS_PAGE_END

END
```



After filling the elements, you can position them with <F6>. The result could be as follows:



- Finally, check correct syntax with <F9>.

The data record created until now should correspond to that in the file STEP\STEP05.PCS.

Summary of step 5

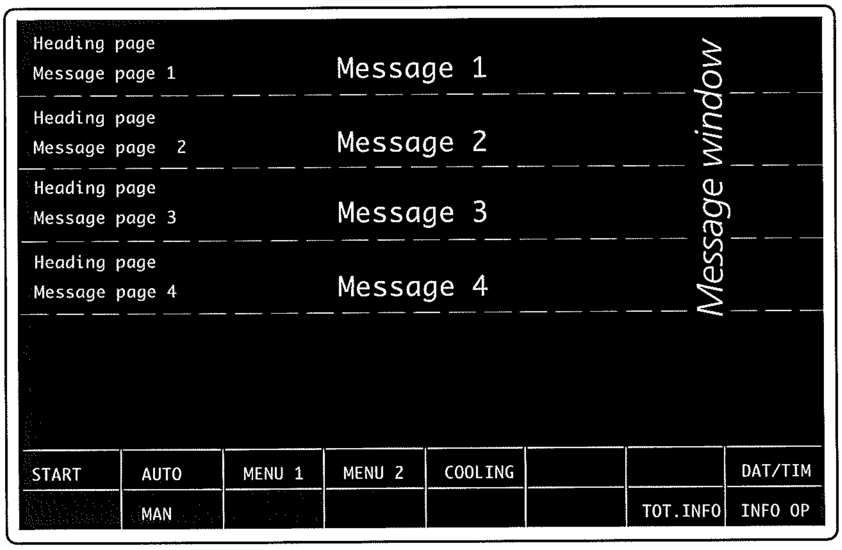
- *Generally, the status page has a size of 640 pixels and is located at the top of the display. The height can be set as required between 0 and 480. A visible status page always covers the idle page (REPORT_PAGE) which is located in the background.*
- *Inserting and positioning elements corresponds to the operations performed for the idle page and the other pages (keys <F1> and <F6>).*
- *The internal variables allow access to information used to adjust or display interface errors, the number of infos, warnings, faults and timeout time.*

Step 6 Create messages

The following operations are based on the state specified in the file *STEP\STEP05.PCS*. The individual messages are displayed in the message window. A message contains a heading page and a message page.

The *heading page* is identical for all messages and defined only once by the key word TITLE. The heading page may only contain static texts and internal variables (information about date, time, number of messages etc.).

On the *message page*, any kind of information is displayed with priorities info, warning and fault. 1024 different message pages can be defined which may contain all static and dynamic elements (texts, graphics, variables).



The message window height (in pixels) is calculated by the formula:

Message window height = (heading page height [pixels] + message page height [pixels] x number of visible messages

For the parameters in the example for global definitions, the following height is calculated:

```
                  Font size of the heading page
                  |    Height of the message page in pixels
                  |    |     Number of visible messages
                  |    |     |
MSG_WINDOW, FONT (2), (20 ), 4
```

results in: *Message window height = (20 pixels + 20 pixels) x 4 = 160 pixels*

- Position the cursor in the line in front of the last "*END*" instruction.

```
SOFTKEY_ROW_END


END
```

Insert and complete the heading page and the 3 message pages:

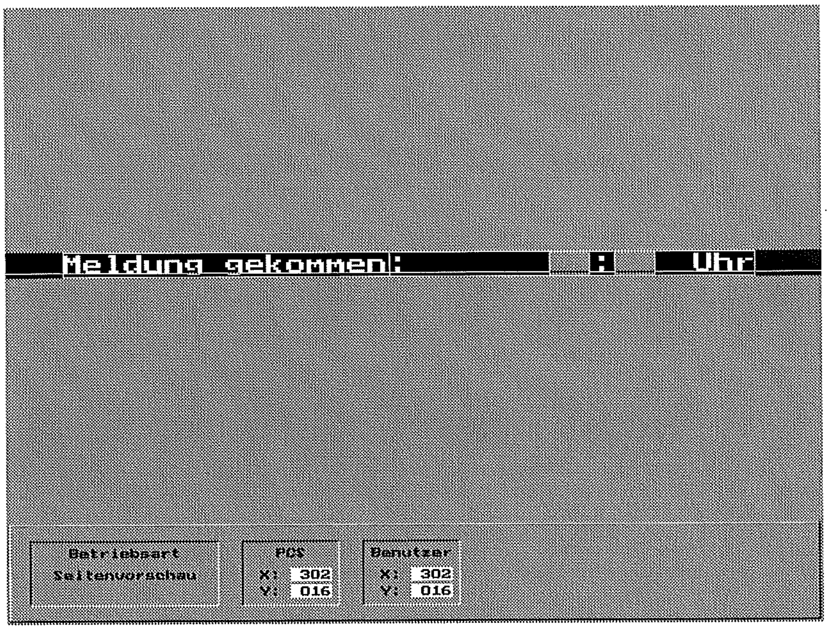
- <F1>
  "Message page" <TAB>
  "as heading page" <ENTER>
- <F1>
  "Message page" <ENTER>
- <F1>
  "Message page" <ENTER>
- <F1>
  "Message page" <ENTER>

```
SOFTKEY_ROW_END

 ———————————— MSG_PAGE, TITEL ————————————————
          Heading page
          |
MSG_PAGE, TITLE

MSG_PAGE_END

 ———————————— MSG_PAGE ————————————————————
          Page number
          |    Priority
          |    |        Clear behaviour
          |    |        |
MSG_PAGE, 0  , INFO   , QUIT

MSG_PAGE_END

 ———————————— MSG_PAGE ————————————————————
          Page number
          |    Priority
          |    |        Clear behaviour
          |    |        |
MSG_PAGE, 1  , WARNING, NO_QUIT

MSG_PAGE_END

 ———————————— MSG_PAGE ————————————————————
          Page number
          |    Priority
          |    |        Clear behaviour
          |    |        |
MSG_PAGE, 2  , FAULT  , QUIT

MSG_PAGE_END

END
```

- Next, insert the static text and the internal variables into the heading and position the elements:

```
STATUS_PAGE_END
 ------------ MSG_PAGE, TITLE -----------------
          Heading page
          |
MSG_PAGE, TITEL
              Position (left bottom corner)
              Xs  Ys           Font size
              |    |           |
INS_ST_TEXT, (45, 19)
                Static text
                |
LANGUAGE (1), "Message come:          :    hour"
                               Position (left bottom corner)
             Designator        Xs   Ys          Font size
             |                 |    |           |
INS_INTVAR, [CLK_HOUR], (428, 19)
                               Position (left bottom corner)
             Designator        Xs   Ys          Font size
             |                 |    |           |
INS_INTVAR, [CLK_MINUTE], (479, 19)

MSG_PAGE_END
```

The heading page could look like the following display:



Meldung gekommen: : Uhr

Betriebsart
Seitenvorschau

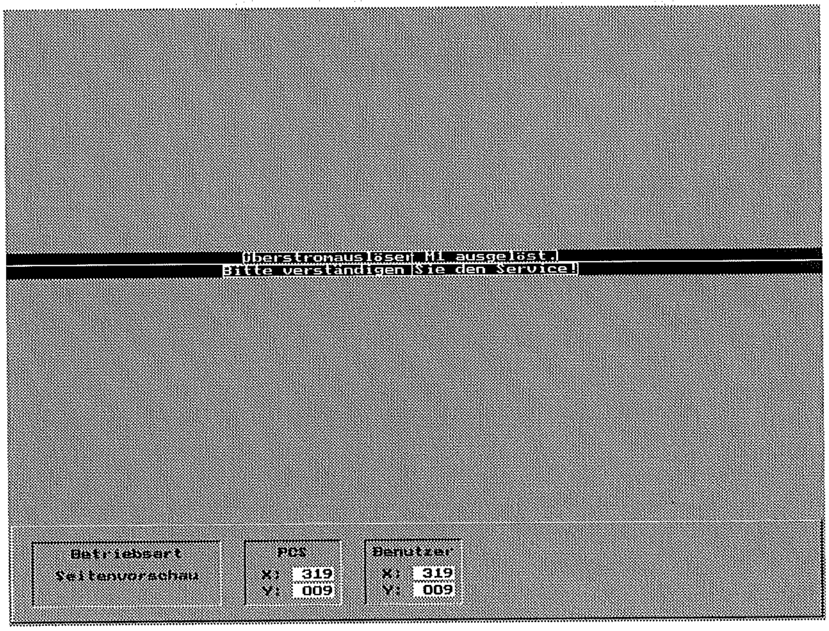
PCS
X: 302
Y: 016

Benutzer
X: 302
Y: 016

Next, insert the messages 0, 1, and 2 :

```
 ───────────── MSG_PAGE ─────────────────────
           Page number
           |     Priority
           |     |         Clear behaviour
           |     |         |
MSG_PAGE, 0  , INFO    , QUIT
               Position (left bottom corner)
               Xs  Ys            Font size
               |   |             |
INS_ST_TEXT, (192, 19), FONT (2)
                 Static text
                 |
LANGUAGE (1), "Close doors!"

MSG_PAGE_END
 ───────────── MSG_PAGE ─────────────────────
           Page number
           |     Priority
           |     |         Clear behaviour
           |     |         |
MSG_PAGE, 1  , WARNING, NO_QUIT
               Position (left bottom corner)
               Xs  Ys            Font size
               |   |             |
INS_ST_TEXT, (152, 19), FONT (2)
                 Static text
                 |
LANGUAGE (1), "Part feeder empty!"

MSG_PAGE_END
 ───────────── MSG_PAGE ─────────────────────
           Page number
           |     Priority
           |     |         Clear behaviour
           |     |         |
MSG_PAGE, 2  , FAULT   , QUIT
               Position (left bottom corner)
               Xs  Ys            Font size
               |   |             |
INS_ST_TEXT, (186, 9), FONT (1)
                 Static text
                 |
LANGUAGE (1), "Overcurrent trip M1 triggered"
               Position (left bottom corner)
               Xs  Ys            Font size
               |   |             |
INS_ST_TEXT, (170, 19), FONT (1)
                 Static text
                 |
LANGUAGE (1), "Please notify the Service!"

MSG_PAGE_END

END
```

E. g. the two static texts of message 3 could be positioned as follows:



Überstromauslöser M1 ausgelöst.
Bitte verständigen Sie den Service!

Betriebsart
Seitenvorschau

PCS
X: 319
Y: 009

Benutzer
X: 319
Y: 009

- Finally, check correct syntax with <F9>.

The data record created until now should correspond to that in the file STEP\STEP06.PCS.

Summary of step 6

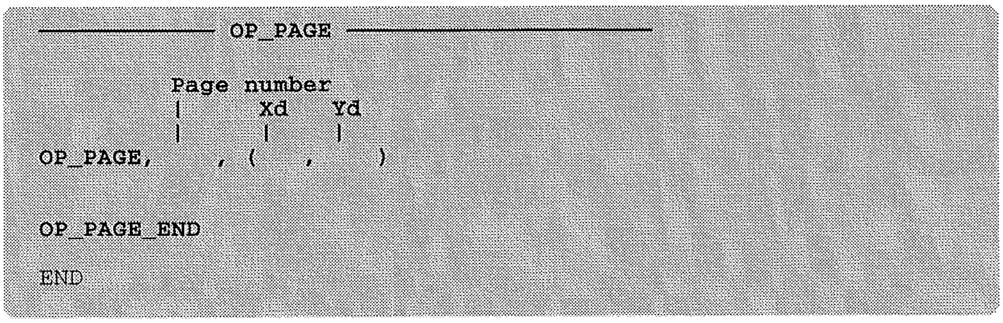
- *Generally, the message page has a size of 640 pixels and is located at the top of the display. The height can be set as required between 0 and 480. A visible message page always covers the idle page which is located in the background.*
- *When determining the global definitions, the height of the heading page is specified by the font, the message page height (in pixels) and the number of messages which are displayed simultaneously.*
- *The messages and the status page are located in one of the four hardware windows. For this reason, the status page or the message page is displayed on PCSmaxi.*
- *The heading page is identical for all messages. A static text (INS_ST_TEXT) and internal variables (INS_INTVAR) can be positioned, to display for example date and time of the message. The internal variable [MSG_TCOME] indicates date and time when the message bit was set in the programmable controller, [MSG_TQUIT] indicates the time when the operator presses CLR and [MSG_TGONE] the time when the message bit is cleared in the programmable controller. [MSG_TQUIT] and [MSG_TGONE] are only relevant with the corresponding clear behavior.*
- *The message page contains information about the message itself. 1024 message pages may be defined according to the message bit. All static (INS_ST_BITMAP, INS_ST_TEXT, INS_RECT etc.) and dynamic elements (INS_VAR, INS_INTVAR) can be used. The dynamic elements may contain up to 8 words. The variable values are frozen.*

Create an operating page for a menu

Step 7

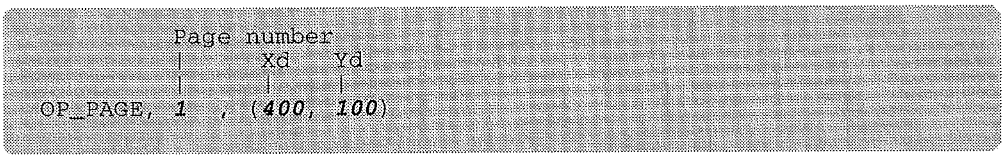
The following operations are based on the state specified in the file STEP\STEP06.PCS.

- Position the cursor in the line before the last "*END*" instruction .
- Insert an operating page description after the messages by pressing <F1>:



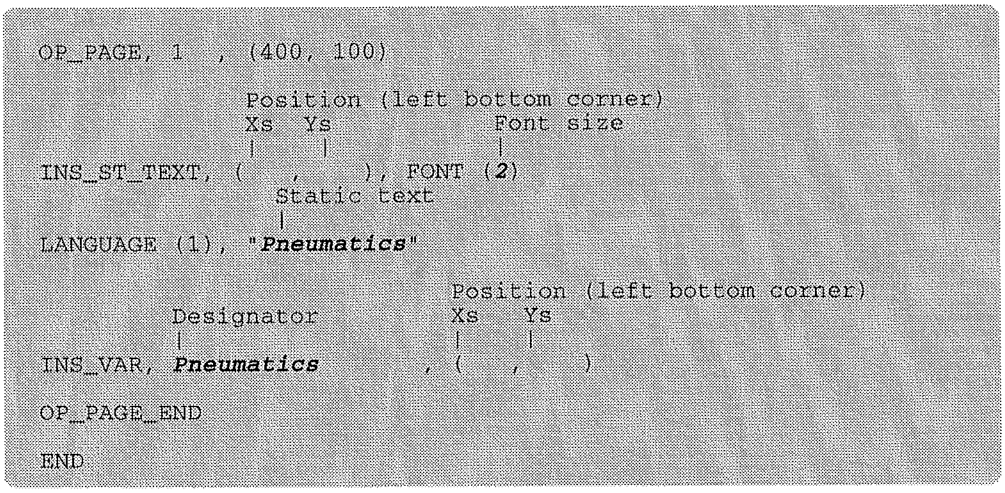
```
———————————— OP_PAGE ————————————————————

          Page number
          |      Xd    Yd
          |      |     |
OP_PAGE,     , (    ,      )


OP_PAGE_END

END
```

Please enter the page number and the size of the window:



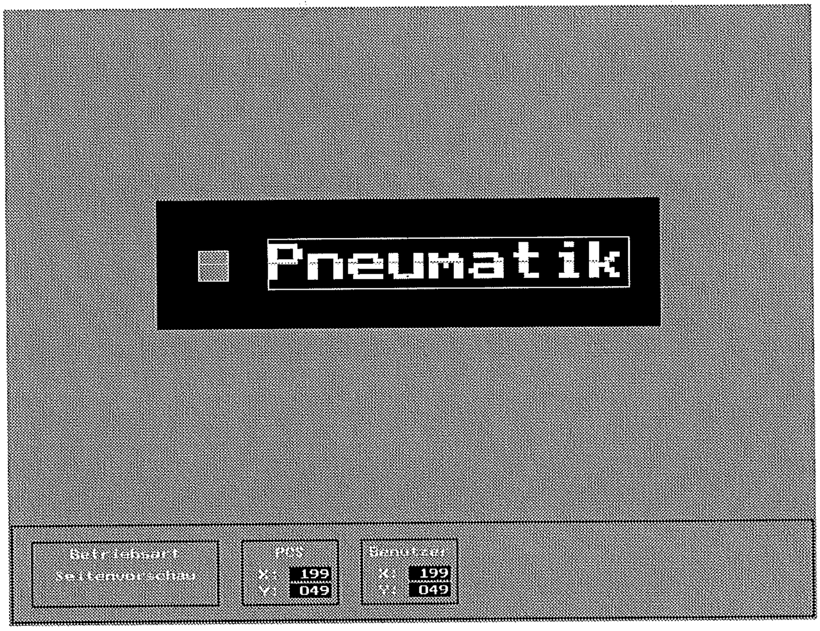
```
          Page number
          |      Xd    Yd
          |      |     |
OP_PAGE, 1   , (400,  100)
```

- Press <F1> to insert the static text command INS_ST_TEXT and a graphic variable INS_VAR and enter all relevant information with the exception of the position data:



```
OP_PAGE, 1   , (400, 100)

                Position (left bottom corner)
                Xs  Ys           Font size
                |   |            |
INS_ST_TEXT, (    ,     ), FONT (2)
                  Static text
                  |
LANGUAGE (1), "Pneumatics"

                             Position (left bottom corner)
          Designator         Xs   Ys
          |                  |    |
INS_VAR, Pneumatics        , (   ,     )

OP_PAGE_END

END
```

In the next step, the text and the variable are successively positioned by pressing <F6>. The result could be as follows:



Betriebsart
Seitenvorschau

PCS
X: 199
Y: 049

Benutzer
X: 199
Y: 049

In the editor, the following position data could be entered:

```
OP_PAGE, 1  , (400, 100)

               Position (left bottom corner)
               Xs  Ys            Font size
               |   |             |
INS_ST_TEXT, (88, 69), FONT (3)
                 Static text
                 |
LANGUAGE (1), "Pneumatics"

                              Position (left bottom corner)
         Designator           Xs   Ys
         |                    |    |
INS_VAR, Pneumatics, (33, 62)

OP_PAGE_END

END
```

- Finally, check correct syntax with <F9>.

The data record created until now should correspond to the one in the file *STEP\STEP07.PCS*.

Summary of step 7:

- *Operating pages for menus have a width between 16 and 640 pixels (adjustable in steps of 16) and a height between 10 and 480 pixels. Menus always cover all windows located underneath (e.g. the idle page, the soft-key labeling, the status page and messages).*

Define a menu — Step 8

The following operations are based on the state specified in the file STEP\STEP07.PCS.

- Position the cursor in the line before the last command END.
- Use the syntax help (<F1> key) to insert the menu description and fill it out.

```
OP_PAGE_END

       Designator Menu number
       |          |
MENU,  Pneumatics, 1
       Operating page number
       |   Position (left bottom corner of the menu in the display)
       |   Xs    Ys
       |   |     |
NODE,  1,  (   ,    )

END
```

Position the cursor in the NODE line and position the menu with the positioning help function (<F6> key).

```
MENU,  Pneumatics, 1

       Operating page number
       |   Position (left bottom corner of the menu in the display)
       |   Xs    Ys
       |   |     |
NODE,  1,  (128, 150)

END
```

- Finally, check correct syntax with <F9>.

The data record created until now should correspond to that in the file STEP\STEP08.PCS.

Summary of step 8:

- *Operating pages are inserted into menus during menu definition and positioned by pressing <F6>. The dimensions of the soft-key labeling and the status line are displayed.*
- *Positioning on the x-axis (0...639) is possible within the range from 0 to 624 in steps of 16.*

Step 9

Create a help page

The following operations are based on the state specified in the file *STEP\STEP08.PCS*.

- Position the cursor in the line before the last command END and insert two help pages by pressing <F1>.

```
NODE, 1, (128, 150)

 ———————————— HLP_PAGE ————————————————————

          Page number
          |    Priority  Xs    Ys     Xd    Yd
               |         |     |      |     |
HLP_PAGE,    ,         , (   ,      ), (    ,     )


HLP_PAGE_END

 ———————————— HLP_PAGE ————————————————————

          Page number
          |    Priority  Xs    Ys     Xd    Yd
               |         |     |      |     |
HLP_PAGE,    ,         , (   ,      ), (    ,     )


HLP_PAGE_END

END
```

Insert static texts and complete the data::

```
 ——————————— HLP_PAGE ————————————————————
          Page number
          |    Priority  Xs    Ys     Xd    Yd
               |         |     |      |     |
HLP_PAGE, 1  , OP     , (128 , 150), (400, 100)

              Position (left bottom corner)
              Xs  Ys            Font size
              |   |             |
INS_ST_TEXT, (  ,    ), FONT (1)
                Static text
                |
LANGUAGE (1), "Toggle with <INC>, <DEC> or <+->"

HLP_PAGE_END

 ——————————— HLP_PAGE ————————————————————
          Page number
          |    Priority  Xs    Ys     Xd    Yd
               |         |     |      |     |
HLP_PAGE, 0  , MSG    , (128, 150), (400 , 100)

              Position (left bottom corner)
              Xs  Ys            Font size
              |   |             |
INS_ST_TEXT, (  ,    ), FONT (1)
                Static text
                |
LANGUAGE (1), "Erase message with <CLR>"

HLP_PAGE_END

END
```

- Position the static texts on the page with the positioning help (<F6> key).
- Finally, check correct syntax with <F9>.

The data record created until now should correspond to that in the file *STEP\STEP09.PCS.*

Summary of step 9:

- *The size and position of help pages can be determined with the positioning help (<F6> key).*
- *For handling the help page, the positioning ranges corresponding to those for a menu are valid: Width of 16 to 640 pixels (adjustable in steps of 16) and height of 10 to 480 pixels.*

Data record transmission

Before transmitting the data record into PCS*maxi*, it must be compiled with <F9>.

Data are transmitted between the PC and the PCS*maxi* programming interface by means of the programming cable PCS 733. Transmission is triggered by <Ctrl-F9>.

Simulation

After data record creation and transmission into the PCS, you can simulate subsequent programmable controller operation directly with the PC or the programmer and check in this way, if the created texts, menus etc. meet your application needs.

The serial PC or programmer interface used for simulation is specified by menu item OPTIONS/INTERFACES.

Simulation requires a connection between the PCS 8010 and the PC by means of the programming cable PCS 733. Simulation is effected by the Lauer driver with 19200 Baud. The rotary switch on the rear of PCS*maxi* must be set to this configuration.

After communication between the PC and the PCS is established, the red COM LED of the PCS extinguishes. You can now change the values of the displayed transfer words and check the PCS response.

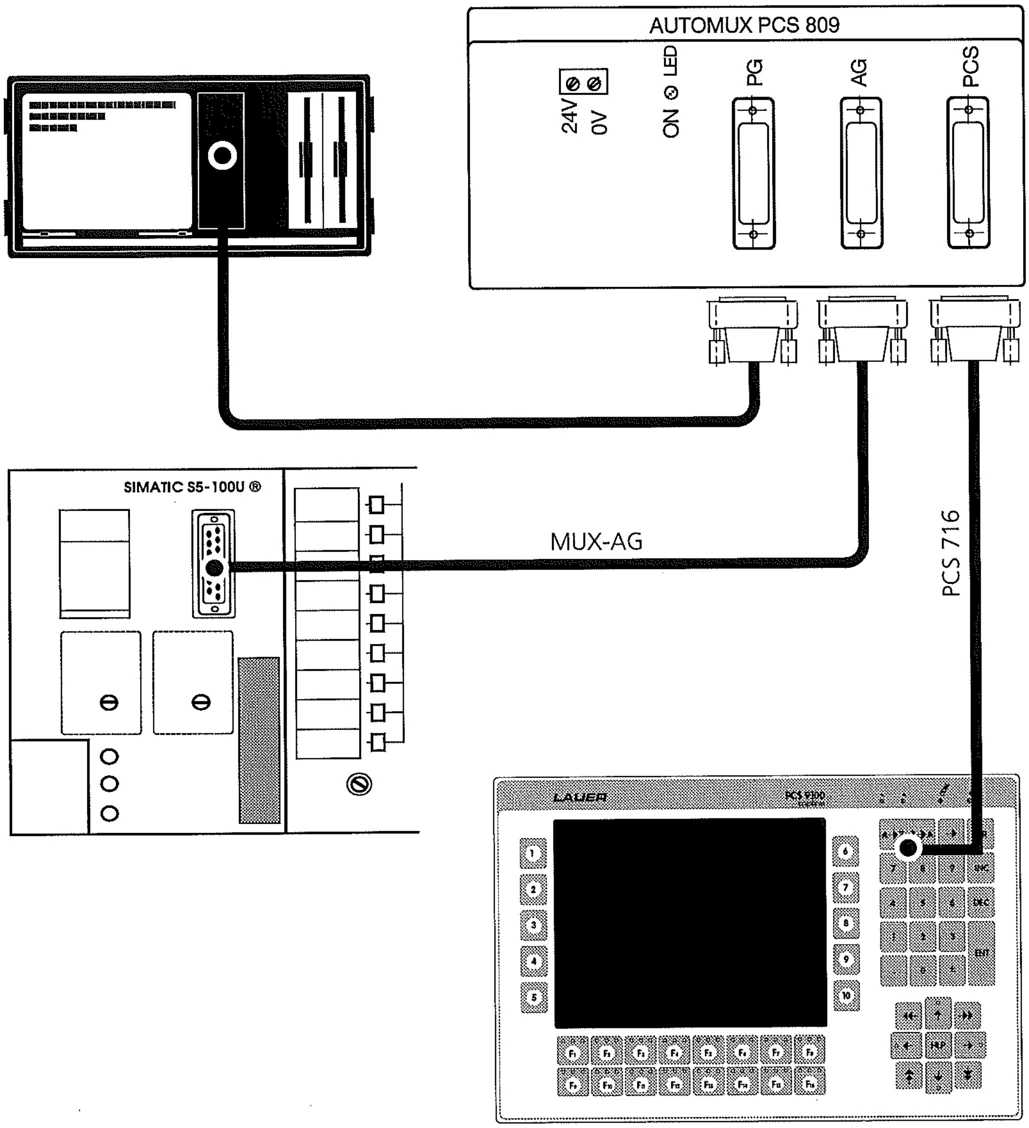
To maintain, as far as possible, an overview of the simulation, all transfer area words are not automatically displayed, but you can select words that should be included or excluded from simulation.

PCS Status (❻)

!	Message/menu disabled
!	Prog. contr. disables message/ menu
?	Operator inquiry, input expected
COM	no communication yet
COM	communication interrupted
SYS	PCS without program or empty cassette plugged in
SYS	Programm transfer (!, ?, COM)

COM, ! = LED static, COM, ! = LED flashing

If communication between the PCS and the Siemens S5 is established via the L1 standard log or the AS511 log, always one programmer interface is activated.



Since small-sized programmable controller systems are equipped with only one programmer interface, limitations have to be considered during startup, i.e. the programmer and the PCS cannot be used simultaneously.

Automux PCS 809 is able to cope with theses limitations. The PCS 809 enlarges the interface between the programmable controller and the programmer so that the controller can be operated simultaneously by the programmer and the PCS. Switching occurs automatically in the MUX.

The PCS 809 is designed as startup tool. After commissioning, the PCS operator console is connected to the programmable controller directly through the programmer interface.

We recommend the Automux PCS 809 to be used, if the PCS*topline* and one of the following programmable controllers supplied by Siemens are to be connected:

S5-90U
S5-95U
S5-100U
S5-115U

Automux PCS 809 is delivered with power cable and adapter cable MUX /AG.

27 Quality and Support

Quality is the most important factor in our company. From the electronic component to the manufactured device, quality is completely tested by qualified personal.

For this purpose, national and international test standards (ISO, TÜV, VDE, CE, Germanischer Lloyd) are applied. Each PCS is tested to 100% at different temperatures (5 ... 55°C) and test voltages (19 ... 33 VDC) and submitted to a permanent test under worst case conditions during 48 hours. This is to assure a maximum of quality!

Our products are not only characterized by a maximum economy and reliability, but also by a comprehensive and complete service.

- Qualified user advice by qualified sales engineers.
- Our support is available to you every day by word and deed. Use our direct info line, if you have questions concerning the PCS*topline*

  Tel ***(+49)(0)7022 / 9660 220 + 221 + 222 + 223***
  Fax ***(+49)(0)7022 / 9660 224***

- Intensive and practice-orientated training for our products. Either in our training center or, after agreement, in your company.
- You do not only receive demo devices, but you are also supported during your first application by our specialists.
- Up-to-date information about our products via „*Lauer aktuell*"
- Update service for our software

From advice to user support, from hotline to service, from manual to training - a comprehensive individual service is guaranteed.

Symbole

! 4, 96
? 4, 96

A

Access class 22, 47
Action 1.0 ... 255 32
Action 2.0 ... 255 32
Action 3.0 ... 255 32
Activation bits 20
Actual value 47
Addition 10
Animation and demonstration 58
ASCII variable 49, 54
Automux PCS 809 97
Autorepeat 10

B

Bargraph 49, 55
Battery 8
Baud rate 41
BCD variable 52
BIN variable 52
Bit pattern 53
BIT variable 50
Bitmap 13
BMP coordinate system 67

C

Character 11
Character set 11
CHARSET 12
Circle 42, 46
COM 4, 96
COMMAND WORDS 18
Commands 33
communication 14
Convert programs 13
Count inputs 8
Create a help page 95
Create a soft-key labeling 78
Create a status page 85
Create an idle page 75
Create an operating page for a menu 92
Create messages 87
Creation of an operating project 65
CSTRING variable 50
Current consumption 8
Cursor 10
Cut out BMP 62

D

Data bits 41
Data record transmission 96
Data storage 8
Data word and data bit assignment 16
Date 24
Date and time 56
Day 24
Day of the week 24
DAY OF THE WEEK, TIME 24
Decrement 10
Define a menu 94
Different firmware modules 58
DIL switch 16
Directory or drive 57
Display 4, 5, 8
Display BMP 62
Display lock 20
DR-DOS 57
Dynamic elements 29, 39

E

ERROR WORD 16
Example for planning a project 69
External dimensions 8
External keyboard 6, 8
External variable 29, 49

F

Fault 38
Files and their contents 58
Filled rectangle 42, 46
Floating output 8
Font 1 11
Font 2 11
Font size 12
Front panel sealing 8
Function keys 4, 5, 31
Functions and tools 28
Fuse 8

G

GBIT 39, 51
GCSTRING 39, 51, 52, 53, 54
Global project data 71
Graphics Workshop 60
GSTRING 39, 51

H

Headline page 35
Help page 38
Horizontal symmetric bar 55
Hour 24

I

Idle page 30
Idle page number 22
INCLUDE 61
Increment 10
Info 38
Information media 29
Interface module 7
Interface settings for printer 41
Internal memory 8
Internal variable 29, 48

K

Key words 16
Keyboard and mouse button assignment 62

L

Language 11
Language selection 11, 12
LANGUAGEMAX 11, 12
Large operating projects 61
Lauer driver 96
LED flashing center 22
LED flashing left 22
LED flashing right 22
LED state center 22
LED state left 22
LED state right 22
Library 13, 59
Line 42, 43, 45

M

MEMMAKER 57
Menu node number 18
Menu number 22
Menu technique 63
Message page 35
Messages 24, 35
Minute 24
Module slots 7
Month 24
Mouse 70
Move the crosshair 62
MS-DOS 57
Multitasking Window System 4, 5

N

Negative horizonal bar 55
Negative vertical bar 55
NODE 37
Noise immunity 8
Numeric keyboard 16

O

OP_PAGE 37
Operating page / menu 37
Operating printer page 26
Operating voltage 8
Operator printer page 39
Overview of the keys used for ASCII variables 54

P

Page descriptions 29, 30
Page number with focus 16
Page preview 62
Paint Shop Pro 60
Paintbrush 59
Parity 41
Pattern 45
PCS 8010 96
PCS status words 16
Pixel 42, 43
Positioning of elements 66, 67
Positive horizonal bar 55
Positive vertical bar 55
Preset value 47
Preset value length 18
Preset value word number 18
Printer connection 6
Printer control sequences 40
Printer interface 41
Priority status 16
Programming 14
Programming cable PCS 733 96
Programming connection 6
Project planning software PCS 9092 57
Protection class IEC 529 8
Pushbuttons 8

R

Real time clock 56
Rectangle 42, 46

S

Scanner 13
Second 24
Simulation 14, 96
Soft-key actions 29
Soft-key labeling 31
Soft-key labeling number 22
Software PCS 9092 57
Specify variables 73
Standard installation 57
Start of external simulation 58
Start of PCS 9092 58
Static bitmap 43
Static bitmap (images, logos) 42
Static elements 29, 42
Static text 39, 42, 43
Status page 34
STATUS WORDS 16
Stop bits 41
STRING variable 50
Subtract 10
Support 98
SYS 4, 96

T

Test standards 98
Transmission direction 16
Transmission lock 18

U

User memory 57

V

Variables formats 54
Variables/Key/Function 10
Vertical soft-keys 5
Vertical soft-keys T1...T10 16
Vertical symmetric bar 55

W

Warning 38
Weight 8
Window 1 9, 30
Window 2 9, 31, 32
Window 3 9, 34
Window 4 9, 37, 38
WINTOOLS 57
WORD 10
WORD variable 49, 53

Y

Year 24

Z

Zoomable 11

- Operating instructions, manuals and software are copyright-protected. All rights reserved. Copy, duplication, translation or transformation of the entire contents or of parts are not allowed. Making a back-up copy of the software for personal use is, however, admissible.
- This manual describes our product PCS 9000 / PCS 9100. The content of this manual is subject to changes without prior notice.
- The programs and data stored on the floppy disk may possibly contain errors or may not be correct. Program execution within the application described in this manual is, however, guaranteed.
- Since floppy disks are data carriers submitted to manipulation, only physical faultlessness is guaranteed. Liability is limited to replacement.
- Improvement ideas or error hints are always welcome.
- These agreements are also valid for the specific appendices to this manual.