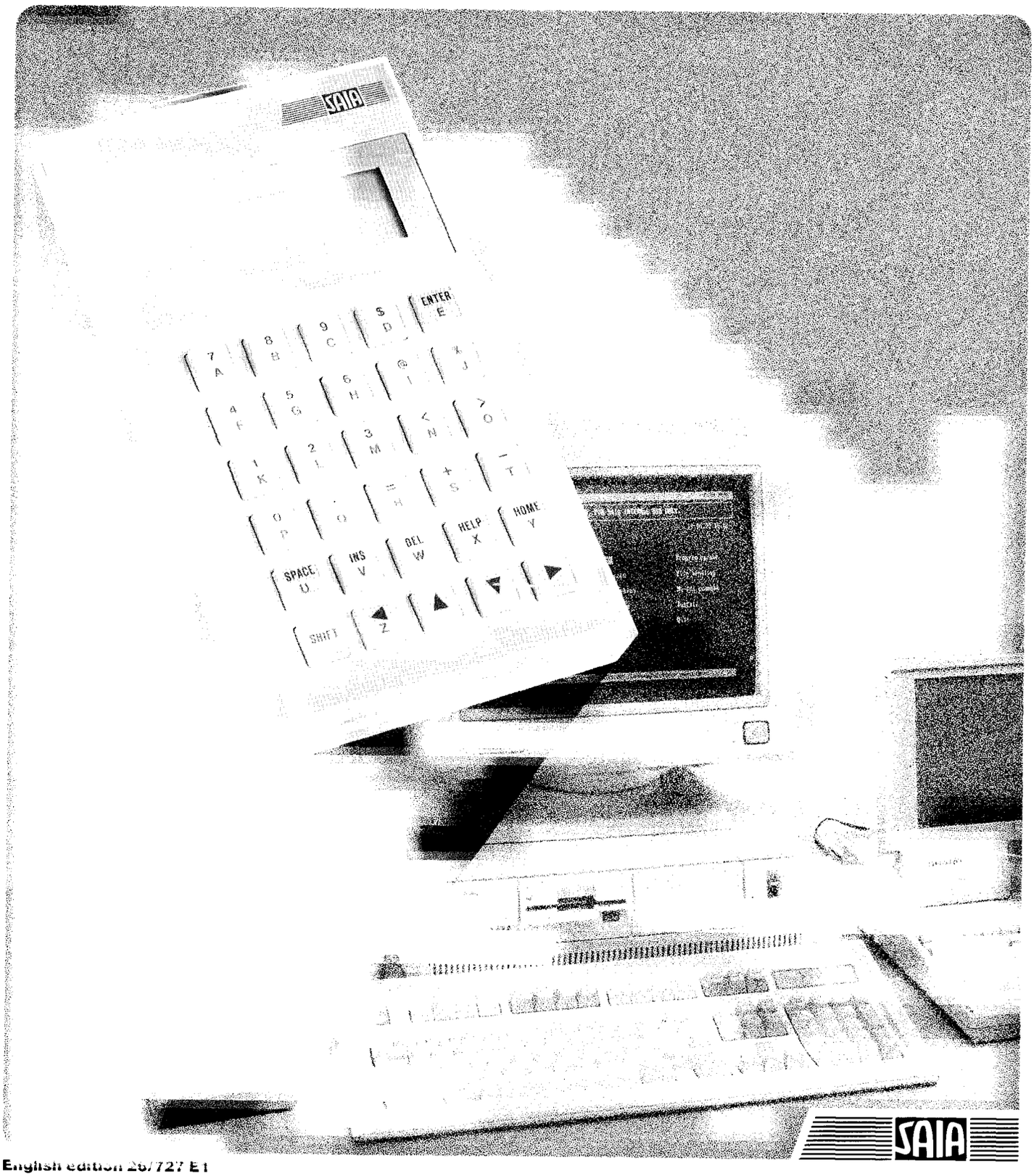
SAIA® PLC
Programmable controllers

Manual
Service and programming unit
PCD8.P100



English edition 26/727 E1

Functional specification
Service and Programming Unit

PCD8.P100

Note:

The chapters 1 - 3 are a introduction, in the other chapters you will find a detailed description.

Subject to change without notice.

Edition 26/727 E1

Price sFr. 40.-

Index

1. Introduction

2. Hardware

2.1 Connection to PCD4 and PCD6 2-1
2.2 Pin assignment of the 25-way D-type connector 2-2
2.3 Keyboard and display 2-3

3. Operation

3.1 Power-up 3-1
3.2 Top-level Menu 3-2
3.3 Menu tree 3-4
3.4 Entering Data 3-12
3.5 Command Memory 3-13
3.6 Password Protection 3-15
3.7 Memory mapping 3-16

4. Commands

4.1 Run Commands 4-18
4.2 Stop Commands 4-19
4.3 Restart Commands 4-20
4.4 Trace Commands 4-21
4.5 Display Commands 4-23
4.6 Write Commands 4-34
4.7 Clear Commands 4-46
4.8 Locate Commands 4-47

5. Self Tests

5.1 Jumpers 5-50
5.2 Power-up Tests 5-50
5.3 Diagnostic Mode 5-52

6. Error Handling

6.1 Error Messages 6-55

1. Introduction

The PCD8.P100 (referred to as the P1) is a portable maintenance and programming unit for local maintenance work and small-scale programming. By means of the four-line, back-lit display, the complete PCD utilities debugger is at the user's disposal. Menu control and help functions make this unit easy to work with, even for the inexperienced maintenance engineer.

For maintenance engineers not entitled to change PLC data, three levels of access are available with a password.

It has the following main characteristics:

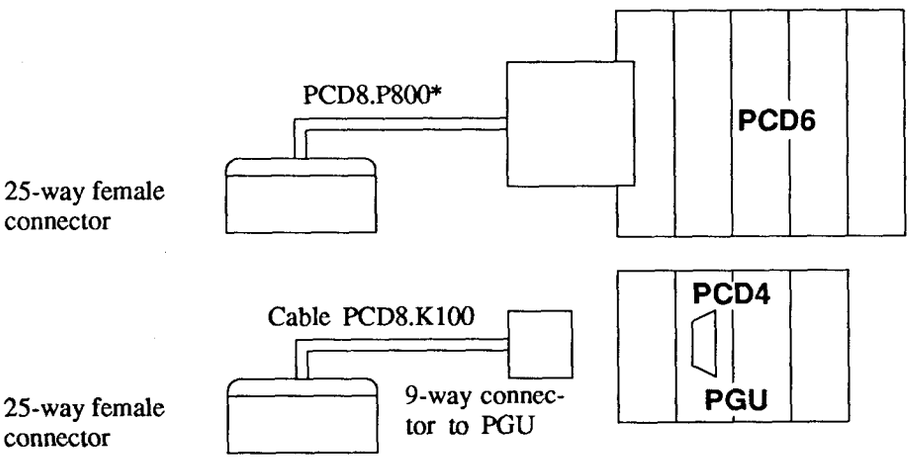
- 4 line x 20 character LCD-Display, with backlight
- Alphanumeric keyboard with 30 keys
- Menu-driven functions. Alternative operation using the alphanumeric keys (as with the PCD utilities debugger)
- Syntax checking for each input, with immediate rejection of incorrect inputs
- "Repeat" command. The last 10 commands entered are stored in memory automatically, enabling the speedy re-execution of previous commands
- Context sensitive help. Over 100 help screens provide quick information on operation if the manual is not to hand
- The P1 allows the display and modification of all elements, the program and the texts, as well as CPU status
- Provides control of all CPU's even in a multi-CPU environment
- Password protection to prevent unauthorized access to PCD data can be programmed
- "Conditional Run" with a breakpoint condition set and single-step processing of one COB only or of all COB's are possible
- Direct connection to the PCD4 via cable and to PCD6 systems via the PCD8.P800 interface

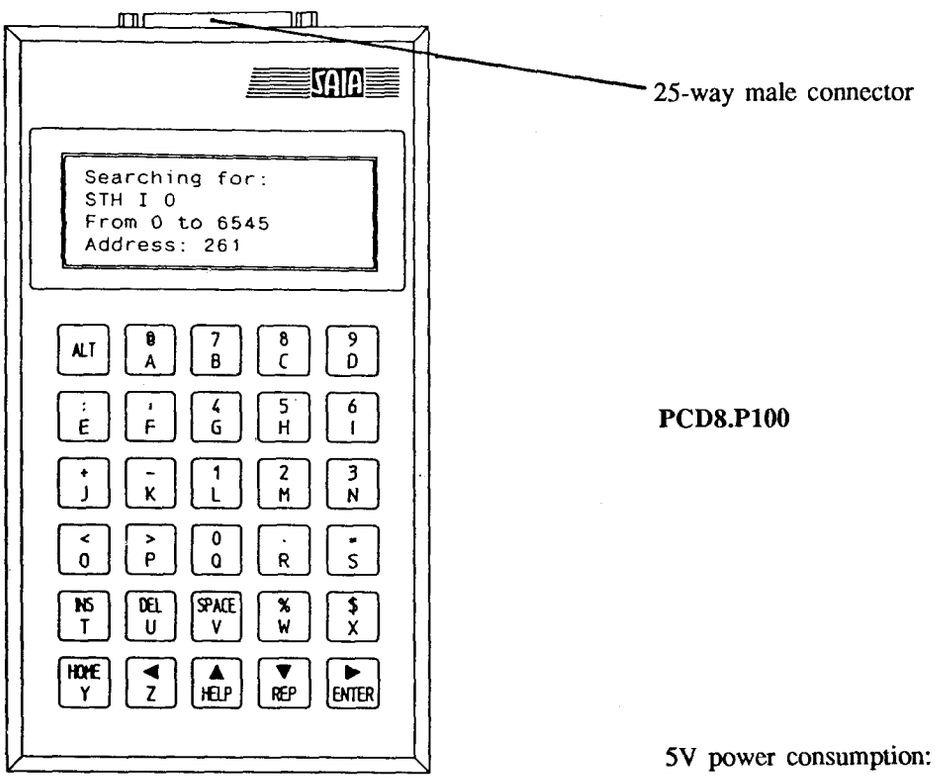
- No internal battery as the P1 is powered from the PCD via the 5V bus
- Automatic power-up hardware tests of the P1

2. Hardware

2.1 Connection to PCD4 and PCD6

The P1 has a standard 25-way male D-type connector. This provides serial connection via an RS-232 interface, and also brings the 5V supply to the unit.





PCD8.P100

5V power consumption: ≤ 120 mA *

* To ensure a good power supply, the PCD8.P800 must be at least revision "A" hardware.

2.2 Pin assignment of the 25-way D-type connector

1	PGN	Protective Ground	
2	TXD	Transmitted data (to PCD's RXD)	
3	RXD	Received data (from PCD's TXD)	
4	RTS	Request to send (P1 output, to PCD's RDYIN)	
5	CTS	Clear to send (P1 input, from PCD's RDYOUT)	
7	SGN	Signal Ground	
12	GND	0V power	Power Supply for P1
13	GND	0V power	
16	VCC	+5V power	
21	VCC	+5V power	

*** WARNING ***

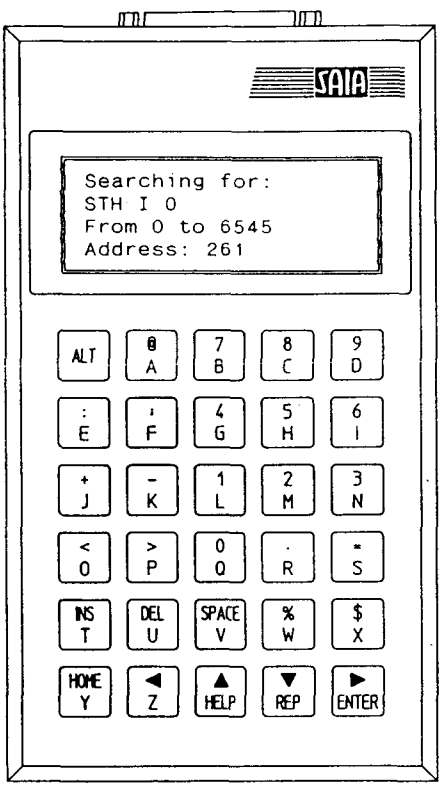
NO OTHER PINS SHOULD BE CONNECTED

2.3 Keyboard and display

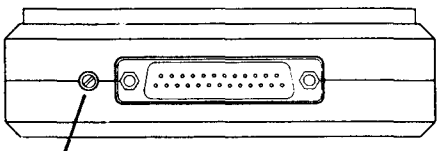
The keyboard has 30 keys, arranged on a 5 x 6 matrix. Each key has two functions, which can be switched with the "ALT"-key. The "ALT"-key is located top left, so that it can easily be depressed with the thumb during hand-held operation.

Key function is normally determined by the menu which has been called. For example, if an alphabetic input is expected, the alphabetic function of the keys is automatically enabled, without having to press the "ALT"-key.

More than one key can be depressed at the same time, and all new key depressions are registered.



The LCD-Display has a backlight to produce optimum contrast whatever the lighting conditions. The contrast and reading angle can be adjusted by inserting a screwdriver into the opening next to the connector.



Contrast adjustment

Key Usage

A...Z	Command input, and for the input of texts, hexadecimal or floating point numbers, and instruction mnemonics
0...9	Numeric input of addresses or values
+ - .	For input of numeric values. "+" and "-" also change the bit order when displaying or writing Inputs, Outputs or Flags
< >	Delimiters in texts for the entry of special characters (in decimal) into texts. The "<" character will move the cursor to the end of an entered value, allowing simple editing with the "DEL"-key. Also used to shift the display left or right when displaying or writing Inputs, Outputs or Flags (ex: < 35 > = #)
=	FB parameter references (= 1, =2 etc)
$ @ : ; %	Special characters often used in texts. ";" is used as a delimiter to separate operands for the "Instruction" command (see section 4.9)
INS	Toggles insert/overtype mode when in "Write teXt", or inserts a line when in "Write Program"
DEL	Deletes characters in "Write teXt", or deletes program lines when in "Write Program"
HOME	Returns to the top-level menu from any position. Also aborts long operations when ** WAIT ** is displayed
ALT	Selects the alternate key function
←	One step to the left in the menu tree
→	One step to the right in the menu tree or selects an option
↑ ↓	One step up or down in the menu tree, or scrolls a display of elements
	All arrow keys move the cursor in some "Write" Menus, they all repeat if held down.

HELP	A specific help text is displayed for each position on the menu, or for each command. There more than 100 different help displays available.
REP	Repeats the previous command. The 10 preceding commands executed commands are stored, see section 3.3. When in "Write teXt", "REP" switches between upper or lower case letters.
ENTER	Enters new data, or is used to confirm or execute an operation
SPACE	Space character, or selects display units, or enters a space and toggles the "ALT"-key state (see section 4.6.5)

3. Operation

3.1 Power-up

When the P1 is connected to the PCD4 or the PCD6/P8, power is applied (5V) and the P1 displays its type and firmware version number for a few seconds. It also begins its power-up tests, in which a "." is displayed as each test is successfully executed. The tests and possible test failure messages are described in section 5.2.

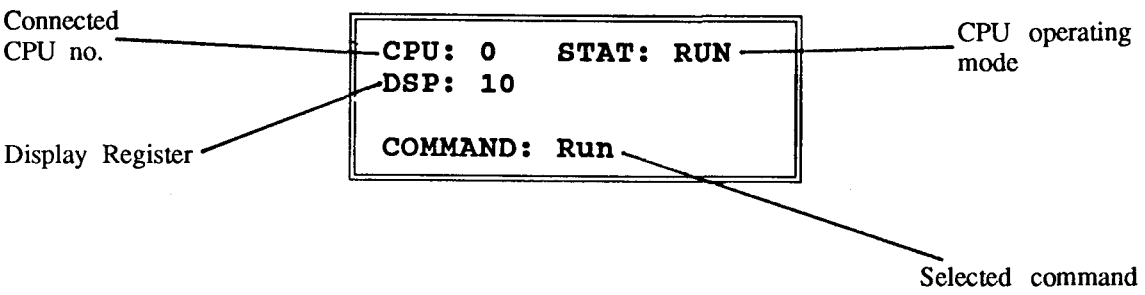
```
SAIA AG, 3280 MURTEN
PCD8.P1          V001
POWER-UP TEST
.....
```

Once the tests have been successfully completed, the top-level menu is displayed.

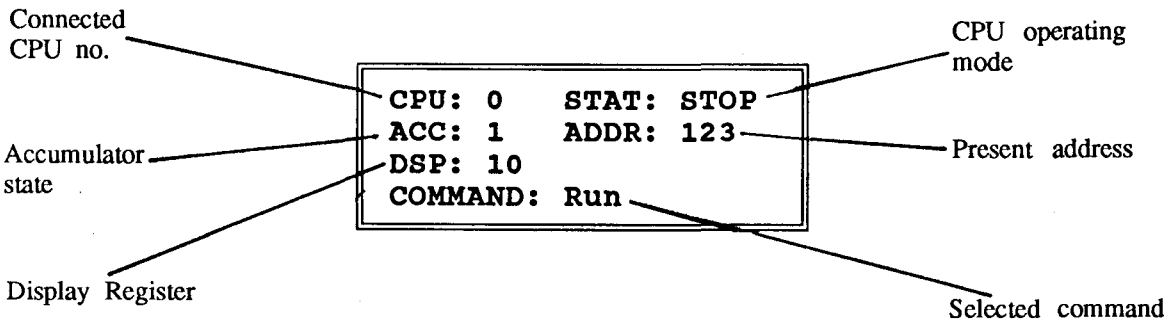
3.2 Top-level Menu

The top-level display has two formats, according to the connected CPU's operating mode.

CPU in RUN or Conditional RUN:



CPU in STOP, Conditional Stop or HALT:



Top-level menu continued

The CPU status and the Display Register value are continually updated on the display, approximately twice a second.

The **CPU status can be one** of the following:

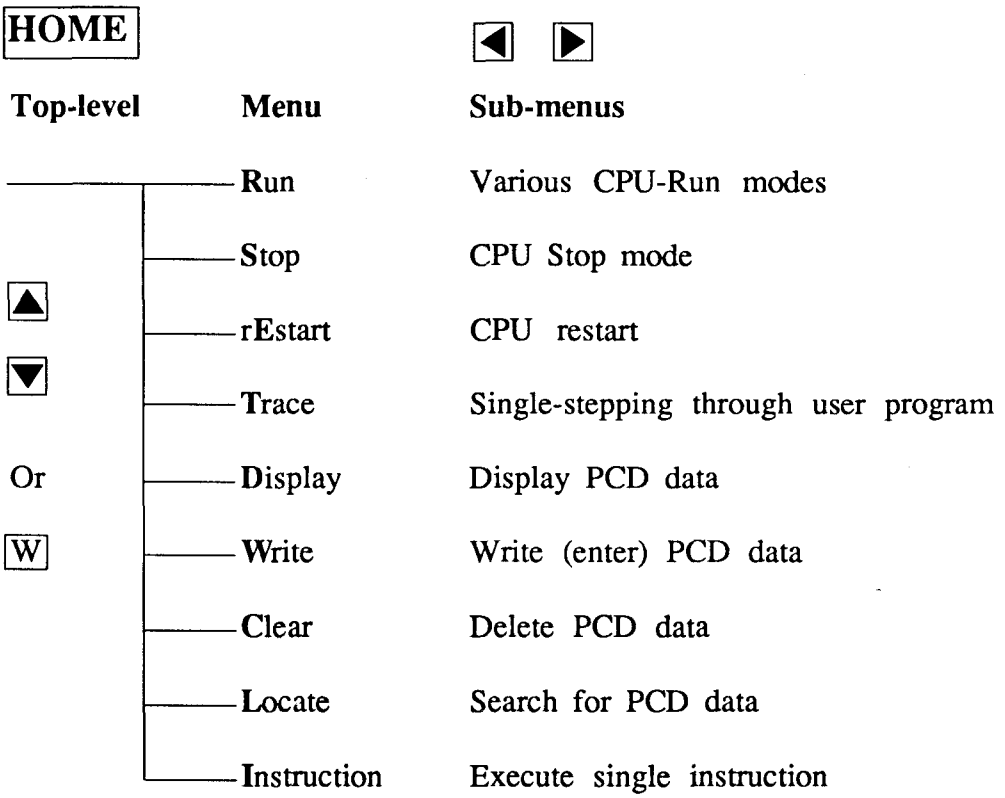
RUN	Run
STOP	Stop
CRUN	Conditional Run (breakpoint is active)
CSTOP	Conditional Stop (breakpoint satisfied)
HALT	Either a HALT instruction in the user program has been executed or a fatal error has occurred
FAULT	The P1 cannot communicate with the PCD, either the cable or the CPU is defective

The **Display Register** can be loaded with the "DSP" instruction. This enables any of the elements (I,O,F,T,C,R) to be permanently displayed in the top-level menu, without having to use the P1.

How to leave the top-level menu is described in detail in the next section (3.3).

3.3 Menu tree

The menus clearly defined in a tree-like structure. At each menu level, the same simple rules apply for the selection of the desired functions. This shows the top-level menu:



There are two ways of selecting a command from the top-level menu:

- by using the arrow keys "↑" and "↓" to step through the commands and selecting one with "→"
- by using the appropriate capital letter to select the command (e.g. "W" for "Write")

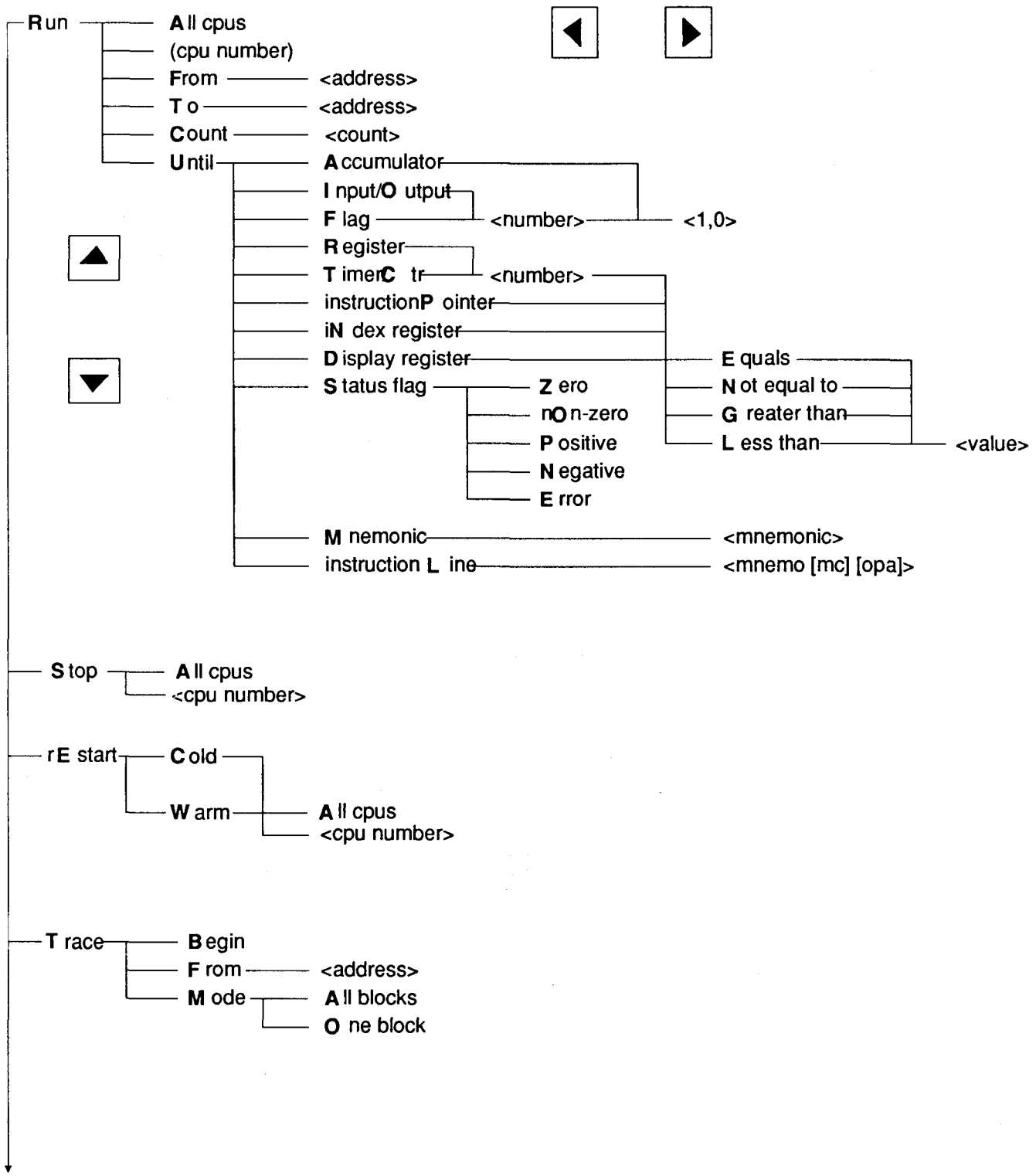
The keys "→" "←" change menu depth to the right or left. To execute a command the menu tree must be traversed from right to left using the "→" key. The procedure is exactly the same for every menu level.

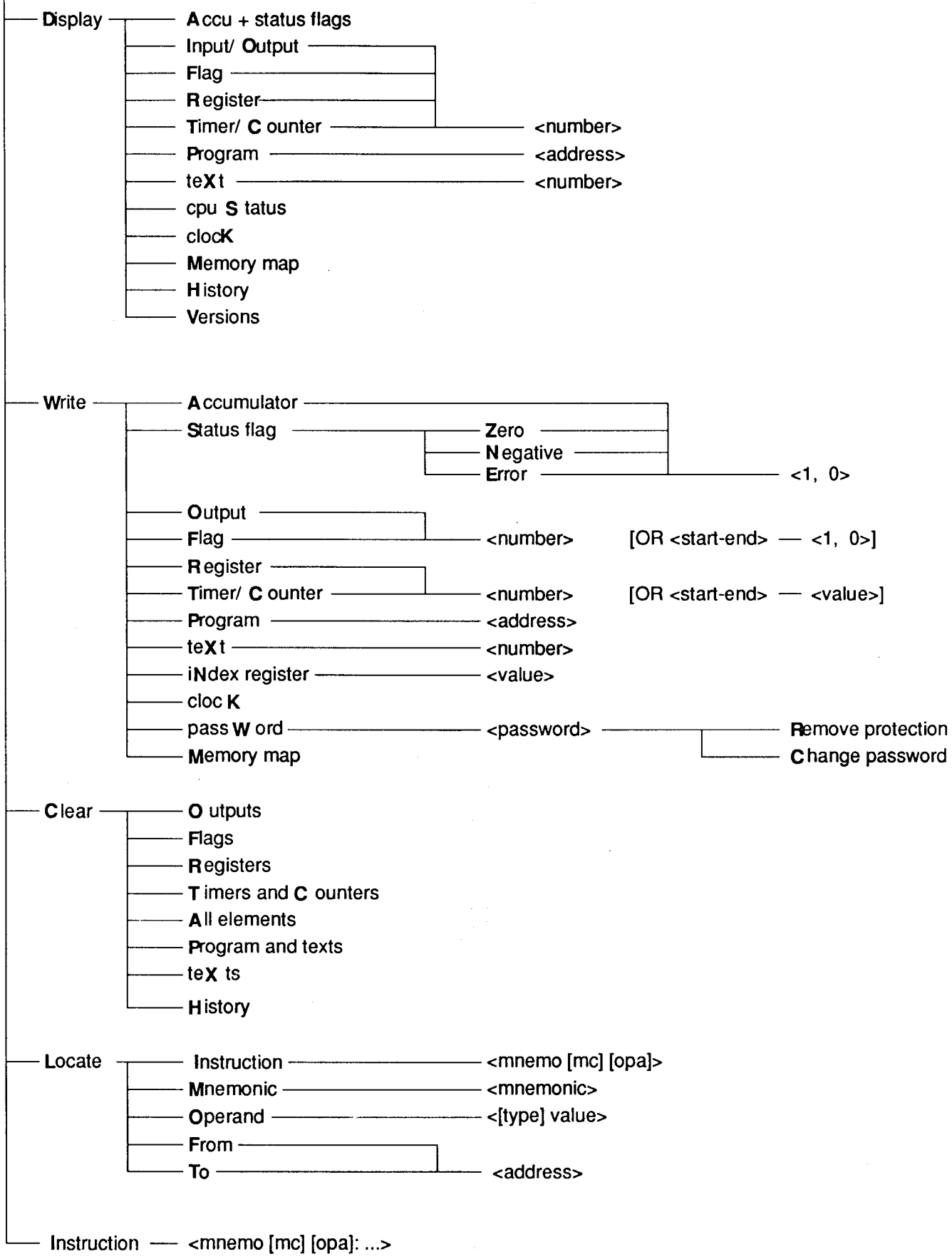
The "HOME"-key allows a return to the top-level menu at any time.

With the "REP"-eat key (ALT-REP), the 10 preceding commands can be recalled, and the recalled command can be executed by pressing "ENTER".

The following two pages show the complete menu tree. With a little practice, these two sheets alone will be enough for you to execute any command. There is always guidance available with the "HELP"-key.

MENU TREE


Run
All cpus
(cpu number)
From
<address>
To
<address>
Count
<count>
Until
Accumulator
Input/Output
Flag
<number>
<1,0>
Register
Timer Ctr
<number>
instruction Pointer
iNdex register
Display register
Status flag
Zero
nOn-zero
Positive
Negative
Error
Equals
Not equal to
Greater than
Less than
<value>
Mnemonic
<mnemonic>
instruction Line
<mnemo [mc] [opa]>
Stop
All cpus
<cpu number>
rEstart
Cold
Warm
All cpus
<cpu number>
Trace
Begin
From
<address>
Mode
All blocks
One block


Display
Accu + status flags
Input/ Output
Flag
Register
Timer/ Counter
<number>
Program
<address>
teXt
<number>
cpu Status
clocK
Memory map
History
Versions
Write
Accumulator
Status flag
Zero
Negative
Error
<1, 0>
Output
Flag
<number>
[OR <start-end> — <1, 0>]
Register
Timer/ Counter
<number>
[OR <start-end> — <value>]
Program
<address>
teXt
<number>
iNdex register
<value>
cloc K
pass W ord
<password>
Remove protection
Change password
Memory map
Clear
Outputs
Flags
Registers
Timers and Counters
All elements
Program and texts
teXts
History
Locate
Instruction
<mnemo [mc] [opa]>
Mnemonic
<mnemonic>
Operand
<[type] value>
From
To
<address>
Instruction
<mnemo [mc] [opa]: ...>

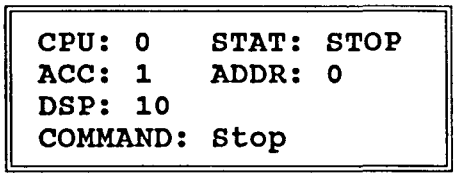
Command Entry Example: Using the arrow keys

This shows the steps taken to enter a "Conditional Run" commmand (Run with breakpoint set):

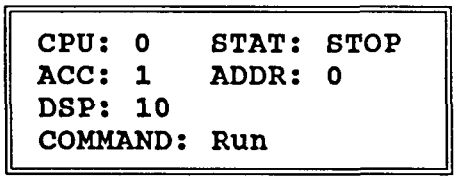
"Run Until Register 10 Equals 1234"

This command is more complicated than most, and serves as a good example of the ease with which complex commands can be entered.

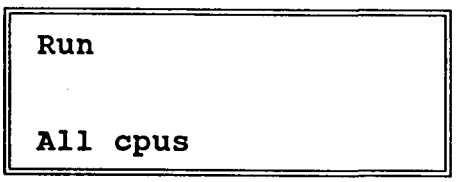
The top-level menu is displayed with the "Stop" command selected. To select the "Run" command, press the "↑" key (see menu tree).



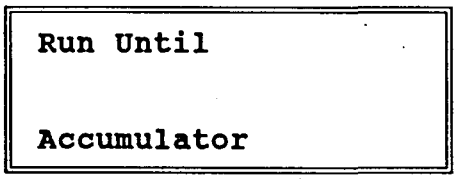
The "Run" command is now displayed on the bottom line. Pressing the "→" key selects the "Run" command and moves right to the next menu level.



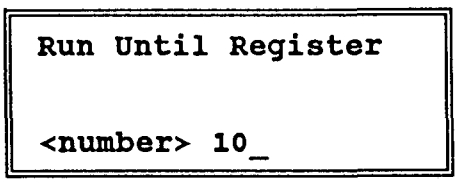
We are now in the first sub-menu of "Run", i.e. "All cpus". Either the "↑" or "↓" key is used to display the option "Until".



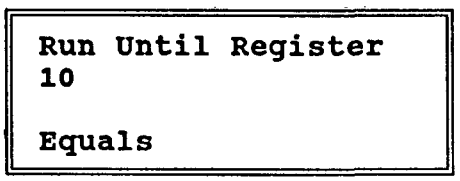
Pressing "→" selects the "Until" option and at the same time displays the next level menu. Again use "↑" or "↓" to display "Register", and select.



Numeric entry of the Register number is now required. Pressing "→" stores the value in memory and at the same time selects to the next menu level.



The first option of this menu is displayed. "Equals" happens to be the very option we want, so only "→" needs to be pressed.



The prompt now requests the Register value. After entering "1234" on the numeric keys (hex or floating point values can also be entered) the whole command is entered and executed by pressing "→".

```
Run Until Register
10 Equals

<value> 1234_
```

The display returns to the top-level menu. The new CPU status is shown: "CRUN" (Conditional Run)

```
CPU: 0    STAT: CRUN
DSP: 10

COMMAND: Run
```

Command Entry Example: Using the command letters

We will enter the same command sequence:

"Run Until Register **10** Equals **1234**"

The top-level menu is displayed, with the "Stop" command selected. To select the "Run" command, press the "R"-key.

```
CPU: 0    STAT: STOP
ACC: 0    ADDR: 0
DSP: 10
COMMAND: Stop
```

The "Run" command is selected directly and the first option of the next menu level is displayed. We want "Until", so "U" is pressed.

```
Run

All cpus
```

"Until" has now been selected. To select "Register", press "R"

```
Run Until

Accumulator
```

Now the Register address can be entered, followed by "→"

```
Run Until Register

<number> 10_
```

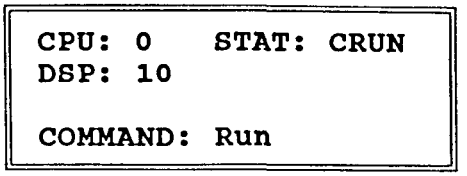
Now we need to select the "Equals" option, so we can press "E" . But, since "Equals" is already displayed, we could also press the right arrow again

```
Run Until Register
10

Equals
```

The prompt now requests the Register value. After entering "1234" via the numeric keys (hex or floating point values can also be entered) the whole command is terminated and executed with "→"

```
Run Until Register
10 Equals

<value> 1234_
```

The display returns to the top-level menu. The new CPU-Status is shown: "CRUN" (Conditional Run).



The keys pressed were:

R U R 1 0 → E 1 2 3 4 → ENTER

It can be seen that this method is considerably faster than using the arrow keys. However, if the command characters have been forgotten, the arrow key method provides a useful alternative.

3.4 Entering Data

The entry of an address or other numeric value is indicated by a menu prompt in angle brackets, ex: "<address>". The cursor appears after the "Prompt", and the numeric keys are activated.

If this point in the menu has been visited before, the value last entered is displayed. A new value can be entered, pressing the first valid key deletes the existing value.

To edit an existing value, the "> " "ALT"-key moves the cursor to the end of the value, and the "DEL" delete key can be used to delete characters to the left of the cursor.

When entering a mnemonic, the alphabetic keys (A-Z) are enabled. To enter a hex or floating point value, where both the alphabetic and the numeric characters are required, the "ALT"-key must be used to select the characters A-Z.

When entering an instruction or operand line, spaces are used to separate the mnemonic from the element type or condition code and the address. Each space entered changes the "ALT" state of the keyboard, alternately selecting the numeric or alphabetic characters so that the "ALT"-key need not be used, see the end of section 4.6.5 for an example.

3.5 Command Memory

Whenever a menu option is selected or a number is entered, it is stored and displayed when the menu is next invoked, together with all preceding entries. If, for instance, the command given in example 3.3 requires a slight amendment, this memory can be exploited:

Previously: "Run Until Register 10 **Equals** 1234"
Now: "Run Until Register 10 **Not equal to** 1234"

Top-level menu, the "Run" command is selected. Select the "Run" menu with "→"

```
CPU: 0    STAT: CRUN
DSP: 10

COMMAND: Run
```

Press "→"

```
Run

Until
```

Press "→"

```
Run Until

Register
```

Press "→"

```
Run Until Register

<number> 10
```

To change from "Equals" to "Not equal to", press "N"

```
Run Until Register
10

Equals
```

If the same Register Value is to be used, press "→", otherwise enter a new value then press "→"

```
Run Until Register
10 Not equal to

<value> 1234
```

To execute the "Run Until Register 10 Not equal to 1234" we only had to press these keys:

→ → → → N →

The **last ten commands executed** are stored in a command list. They can be recalled and displayed by pressing "ALT" "REP" (hold the "ALT"-key down and press "REP")

After execution of the preceding command sequence, the top-level menu shows the CPU is in "CRUN" mode with the "Run" command selected

```
CPU: 0    STAT: CRUN
DSP: 10

COMMAND: Run
```

"ALT" "REP" displays the last command entered. To display the command entered before that, press "ALT" "REP" again.

```
Run Until Register
10 Not equal to

<value> 1234
```

This earlier command can be re-executed by pressing "→"

```
Run Until Register
10 Equals

<value> 1234
```

The "REP"eat-key can be used from any menu, not only from the top-level menu.

3.6 Password Protection

Four levels of security are available to prevent the P1 being used by unauthorized personnel to change element values, the program or the CPU's operating mode.

A pre-programmed password must be entered to remove the protection (see section 4.6.9).

In all levels of protection, the "Display" and the "Locate" commands are enabled.

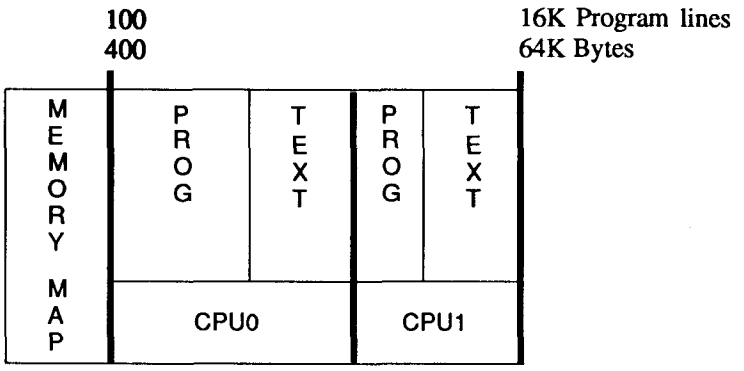
PROGRAM PROTECTION LEVEL	WIRTE (CHANGE)			READ
	PROG AND TEXT	CPU STATUS	ELEMENTS O/F/T/C/R/CLK	ALL FUNCTIONS
0	YES	YES	YES	YES
1	NO	YES	YES	YES
2	NO	NO	YES	YES
3	NO	NO	NO	YES

YES: Function available
NO: Function disabled

- Without entering a password, the programmed protection level cannot be changed
- By entering protection level 0 and a password, all functions are available. However, unauthorized persons are not able to enter the password
- Password and protection level are identical for all CPU's in the same PCD-System (rack)
- The password is stored in CPU-Memory (not in the P100) (Details see chapter 4.6.9)

3.7 Memory mapping

The user memory is divided into program and text for each CPU. Preceding these is the "Memory map", in which memory layout is stored. This "Memory map" occupies 100 program lines of CPU 0. For a PCD with two CPU's, it looks like this:



For each CPU, it is possible to allocate any amount of program and text up to the maximum memory capacity as desired, using the "Write Memory map" command.

WARNING: Each change to the "Memory map" completely deletes all code and text in memory.

When a new PCD-System is commisioned (PCD4 and PCD6) or when the backup battery has been changed, the system program automatically allocates memory for each CPU present:

CPU 0	Code	9K	lines
	Text	4K	characters
CPU 1..6	Code	8K	lines
	Text	4K	characters

In this way, the user entering the system does need to concern himself with the "Memory map" and can immediately start programming.

4. COMMANDS

4.1 Run Commands

The "Run" command can set any CPU into Run mode, and it begins to execute instructions. The connected CPU can be supplied with an optional start address or breakpoint.

Run All cpus

Puts all CPUs in the rack into Run mode.

Run <cpu number>

Puts the specified CPU (0-6) into Run mode.

Run From <address>

Puts the connected CPU into Run, first setting the instruction pointer to <address>. The address must be inside the present block (COB, XOB, PB, FB, SB, ST or TR). It is not possible change the block being executed with this command.

Run To <address>

Sets a breakpoint at <address>. The connected CPU is placed into Conditional Run mode. It will go into Conditional Stop when the specified address is reached.

Run Count <count>

The connected CPU goes into Conditional Run mode, and executes <count> instructions, before going into Conditional Stop.

Run Until .....

Sets a breakpoint and places the connected CPU into Conditional Run mode. When the breakpoint condition is satisfied, the CPU is placed into Conditional Stop.

The breakpoint can be on the value of any element, the Accumulator, the Instruction Pointer (same as "Run To <address>"), the Index Register, the Display Register, a Status Flag (Zero, Positive, Negative, Error), or when a particular instruction is executed (specified either by the mnemonic only, or by the entire instruction line including the operand). See the command map in section 3.6 for details of the "Run Until..." command structure.

*** NOTE ***

When a CPU is in Conditional Run the "Run" LED will be blinking.

4.2 Stop Commands

A single CPU, or all CPUs can be placed into Stop mode.

Stop All cpus

Places all CPUs in the rack into Stop mode.

Stop <cpu number>

Places the specified CPU (0..6 for PCD6, 0 or 1 for PCD4) into Stop mode.

4.3 Restart Commands

The Restart command is the equivalent of a system reset. Either all CPUs, or an individual CPU (1..6) can be restarted. It is not possible to restart only CPU 0 because this is the master CPU.

After a restart, the CPU remains in Stop mode. If all CPUs are restarted, all CPUs will remain in Stop.

The "Restart" command has two options, "Restart Cold" or "Restart Warm":

```
Restart Cold All cpus
Restart Cold <cpu number>

Restart Warm All cpus
Restart Warm <cpu number>
```

"Restart Cold" simulates a power up of the CPU, the CPU's internal data tables are set up, and the Instruction Pointer is set to the beginning of the start-up Exception Organisation Block 16 (XOB 16), if present. XOB 16 is therefore executed when the CPU is subsequently put into Run mode, before COB 0 is executed.

"Restart Warm" initialises the CPU's internal data tables, but sets the Instruction Pointer to the beginning of COB 0. XOB 16 is NOT executed when the CPU is put into Run.

A "Restart" must be done to recover a CPU from the Halt state, or if the program structure has been changed through the "Write Program" or "Write teXt" commands.

Before the restart is done, the command must be confirmed by typing "Y" (ALT-Y). Pressing any other key aborts the command:

```
rEstart Cold All
cpus
ARE YOU SURE ?
PRESS ALT-Y IF YES
```

After a restart, the following elements are cleared (set to zero):

- Accumulator and the Status Flags Z, N, E (P=1).
- All Outputs.
- All Volatile Flags, see "DEFVM" instruction.
- All Timers.
- Index Register.
- Trace mode is set to "All blocks", see section 4.4.

The following elements values are NOT altered by a restart:

- Registers and Counters.
- Display Register.
- Non-volatile flags.

4.4 Trace Commands

The "Trace" command allows the program in the connected CPU to be single stepped, executing one instruction at a time. The state of the Accumulator, Status Flags and Index Register are displayed. "Trace" can only be done if the connected CPU is in Stop or Conditional Stop.

All blocks (COBs and XOBs) can be stepped, or stepping can be confined to the current block. It is also possible to skip the stepping through any FBs, PBs or SBs which are called.

Due to user interface differences, the operation of "Trace" is slightly different to that of SBUG's "Trace", although the same operations are available.

"Trace" command options:

Trace Begin

Enters Trace mode, at the present instruction pointer address. No instructions are executed until the down or right arrow key is pressed. The next instruction is displayed.

Trace From <address>

Enters Trace mode in the same way as "Trace Begin", but a start address can be entered. This address must be inside the current block (COB, XOB, PB, FB, SB, ST or TR). This command can be used to skip a section of code inside a block.

Trace Mode All blocks

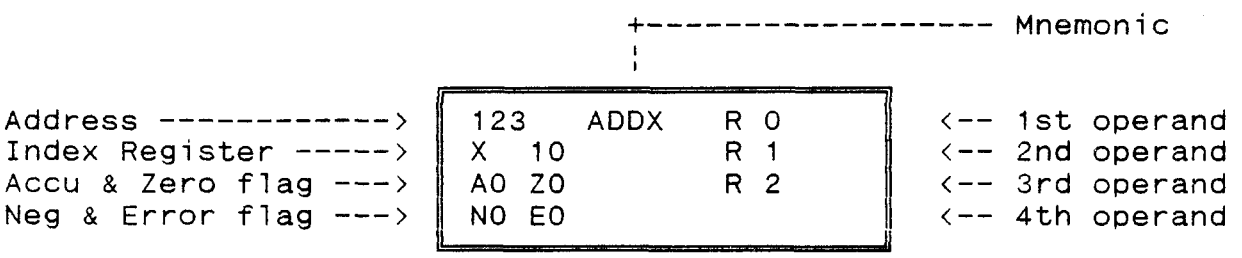
Selects the operating mode for Trace. This enables single stepping through all the existing COBs. At the end of an XOB or COB, the next COB is begun. This is the default mode.

Trace Mode One block

This command selects the operating mode for Trace. It enables the single stepping through instructions within the current COB or XOB only. When the end of block instruction is reached (or NCOB instruction), the next COB is NOT executed, stepping continues at the start of the current COB/XOB.

NOTE: A "Restart" will set the trace mode back to "All blocks".

The Trace display



The display shows the present Instruction Pointer, the Index Register value, the Accumulator and the Status Flags. The displayed instruction has not yet been executed, therefore the Index Register, Accumulator and arithmetic status Flag values are the values BEFORE execution of the instruction.
======

Since the display has 4 lines, instructions with up to 4 operands can be displayed. If the instruction has more than 4 lines, as may be the case with a CFB parameter list or an ST/TR input/output list, then only the first 4 operands can be displayed. To display these operands the "Display Program" command must be used.

These keys are used to control the stepping:

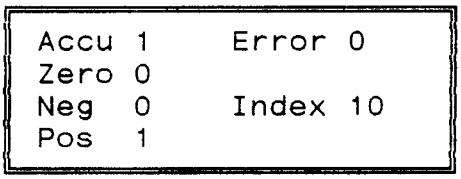
Down Arrow	Executes the displayed instruction. If the instruction is a block call CSB, CFB or CPB, the block is excuted but is NOT single stepped (the instructions are NOT displayed). The trace resumes at the next instruction after the call. This is the equivalent of SBUG's "Trace No-calls" command.
Right Arrow	Executes the displayed instruction. If the instruction is a block call CSB, CFB or CPB, the call is executed, and the block IS single stepped, each instruction is displayed.
Left Arrow	Exits Trace mode, returning to the Trace menu.
HOME	Exits Trace mode, returning to the top-level menu.

4.5 Display Commands

4.5.1 Display Accu + status flags (and Index Register)

This command is available only when the connected CPU is in Stop, Conditional Stop or Halt.

The current value of the Accumulator, Status Flags and Index Register are displayed on one display:



In reality, the "Pos" positive flag does not exist in the CPU, it is always taken as the inverse of the "Neg" negative flag.

Press the left arrow or HOME key to exit.

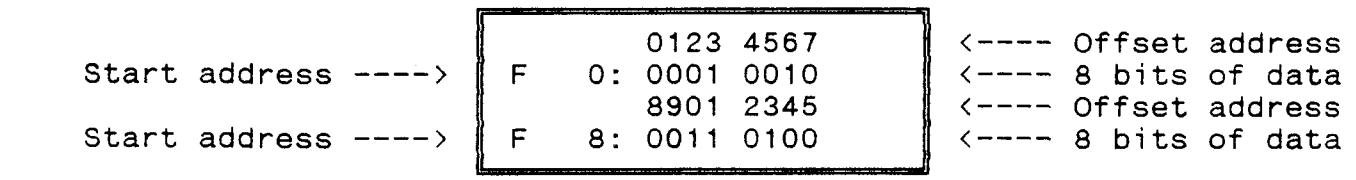
4.5.2 Display Input/Output and Display Flag

The "Display Input/Output" and the "Display Flag" commands are functionally the same.

The <address> of the first Input/Output or Flag to be displayed must be entered.

The display can be scrolled up and down, left and right, the order of the display can be reversed (least significant bit first), or the units of display can be switched between binary and hexadecimal/bcd (binary coded decimal). The display is continually refreshed.

The display shows the addresses, and 16 I/O/F values at a time:



These keys control the display:

Down Arrow	Scrolls the display down, displays the next 8 elements.
Up Arrow	Scrolls the display up, displays the previous 8 elements.
Left Arrow Right Arrow	These shift the display one DIGIT to the left or to the right. If hex/bcd units are selected, the display is shifted by one hex/bcd digit (4 bits).
< >	Scrolls the display left or right by one BIT. This also works when the display units are hex/bcd, but the value of the preceding element becomes the MS (or LS) bit of the first (or last) hex/bcd digit. (The left and right arrow keys shift the display by one DIGIT, not one bit).
SPACE	Selects the display units, binary or hexadecimal/bcd. The hex/bcd displays are the same. Bits are grouped in sets of 4 and displayed as a hex digit (0-F).
+ -	'+' changes the order of the displayed bits to ascending order, least significant element first. '-' changes the order of the bits to descending order, most significant bit first. This also works when the display units are hex/bcd. This is useful for bcd inputs read using the BITIR or DIGIR instructions.
HOME	Exits and returns to the top-level menu.

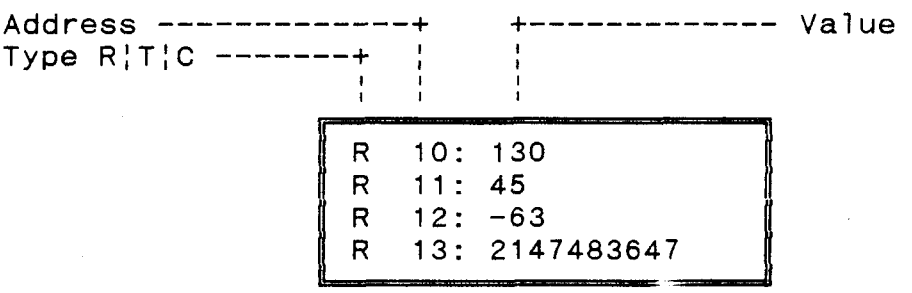
4.5.3 Display Register and Display Timer/Counter

The "Display Register" and the "Display Timer/Counter" commands are similar.

The address of the first Register or Timer/Counter to be displayed must be given.

The display can be scrolled up and down, and the display units can be selected as decimal, hexadecimal, binary, floating point or ASCII by pressing the SPACE key. (Floating point and ASCII units are for Registers only). The display is continually refreshed.

The display shows the R/T/C number, up to 4 values are shown at a time:



These keys control the display:

Down Arrow	Scrolls the display down.
Up Arrow	Scrolls the display up.
SPACE	Selects the display units, either decimal/bcd (the default), hexadecimal, floating point (Registers only), binary, or ASCII (Registers only). If binary units are selected, only one R/T/C value can be displayed at a time. The selected units are easily determined from the display format. See the examples on the next page.
Left Arrow	Returns to the prompt to enter the R/T/C number. This allows fast selection of another R/T/C without having to press the up/down arrow keys many times.
HOME	Returns to the top-level menu.

Display Register and Timer/Counter - Examples

DECIMAL/BCD UNITS:
Values are left-justified, without leading zeros, see the example on the previous page.

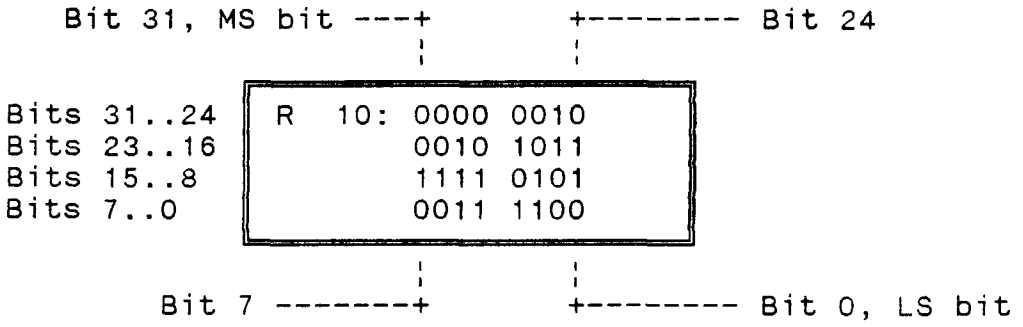
HEXADECIMAL UNITS:
8 digits with leading zeros are displayed. Values are postfixed with "H".

```
C    0: 00000082H
C    1: 00AB002DH
C    2: 0000003FH
C    3: FFFFFFFFH
```

FLOATING POINT UNITS:
Timers and Counters cannot be displayed in floating point units, because they cannot contain floating point values. Floating point values contain a decimal point and an exponent. Invalid floating point values are displayed as "?.??????E+??".

```
R1000: 1.234567E+01
R1001: 0.000000E+00
R1002: ?.??????E+??
R1003: -2.414872E-12
```

BINARY UNITS:
Only one value is displayed at a time. The most significant bit is shown first (bit 31). Bits are grouped in blocks of 4.



ASCII UNITS:
For Registers only. Each Register can contain up to 4 x 8-bit ASCII characters. The four characters are displayed inside inverted commas, with the most significant character first (bits 31..24) and the least significant character last (bits 7..0). Non-displayable control characters, or characters with codes greater than 127, are displayed as a blank space.

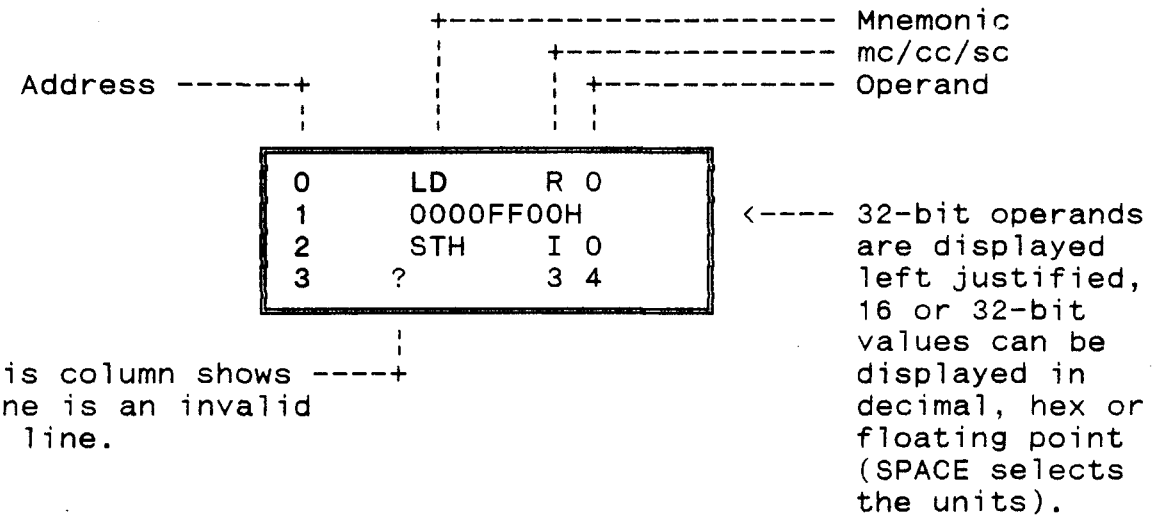
```
R    0: '   a'
R    1: '   b'
R    2: 'C   '
R    3: '1234'
```

4.5.4 Display Program

Disassembles and displays program lines from the code within the connected CPU. The start address must be given. The display can be scrolled up and down to show the next or previous instructions.

If a "?" character appears after the address, then the instruction line is invalid. This can be an invalid operand, an unexpected operand, a missing operand or an invalid opcode.

The display shows the addresses and 4 instruction or operand lines:



These keys control the display:

Up Arrow	Scrolls the display up, displays the preceding instruction line.
Down Arrow	Scrolls the display down, displays the next instruction line.
SPACE	Selects units for the display of LD, LDH and LDL values. For LD (32-bits) the units can be decimal (the default), hex or floating point. For LDH/LDL, the units can be decimal or hex.
Left Arrow	Returns to the prompt to enter the address. This allows fast selection of another instruction line without having to press the up/down arrow keys many times.
HOME	Returns to the top-level menu.

*** NOTE ***

Inputs and Outputs share the same addressing space. The P1 does not know if an address references an Input or an Output. Therefore both Inputs and Outputs are indicated with an "I" as a medium control code (except where the instruction must access an Output). The same is true for Timers and Counters, the "C" medium control code is used for both.

4.5.5 Display teXt

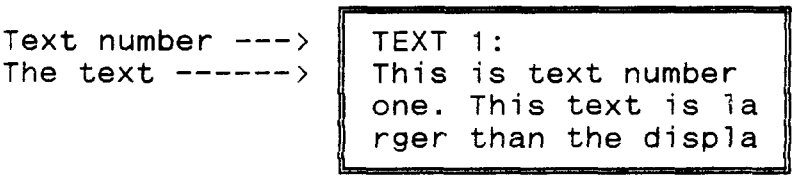
One text is shown on the display at at time. The starting text number must be entered.

Texts longer than 20 characters wrap around onto the next line.

A text which is larger than the display can be scrolled left/right and up/down using the arrow keys. When the start of end of text is reached, the up/down arrow keys select the next or the previous text.

Text units are ASCII only. Non-displayable characters are displayed as decimal values enclosed in angle brackets "<12>".

The first line of the display shows the text number, leaving three lines for the text display window. The end of the text is indicated by the character "■".



These keys control the display:

Up Arrow	Scrolls the text up one line (20 characters). If at the start of the text, the preceding text is displayed.
Down Arrow	Scrolls the text down a line (20 characters). If at the end of the text, the next text is displayed.
Left Arrow	Scrolls the text left, if not already at the start of the text.
Right Arrow	Scrolls the text right, if not already at the end of the text.
HOME	Returns to the top-level menu.

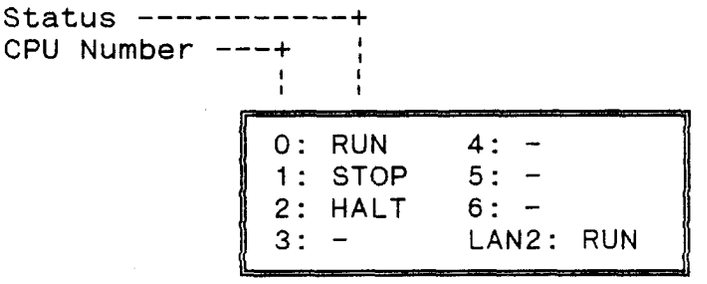
4.5.6 Display cpu Status

Shows the present status (Run, Stop etc) of all CPUs and the opticnal LAN2 co-processor.

For the PCD6, up to 7 CPUs and a LAN2 can be present. For the PCD4 either one or two processors, or one processor and a LAN2 can be present.

The status texts are as described in section 3.2. If a CPU is not connected, a dash is shown "-".

PCD6 display example:



(CPUs 3..6 are not present)

PCD4 display example:

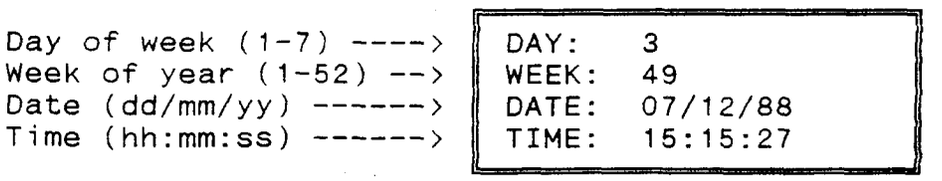
(2 CPUs present)

(1 CPU and a LAN2 present)

4.5.7 Display clocK

Displays the PCD's real time clock. The display is continually refreshed, so the date and time is always valid.

All CPUs in the same rack share the same real time clock.

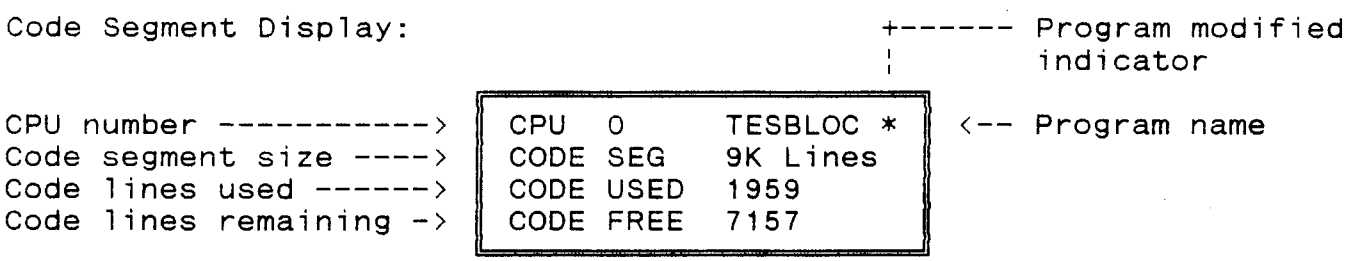


Day 1 = Monday
Day 7 = Sunday

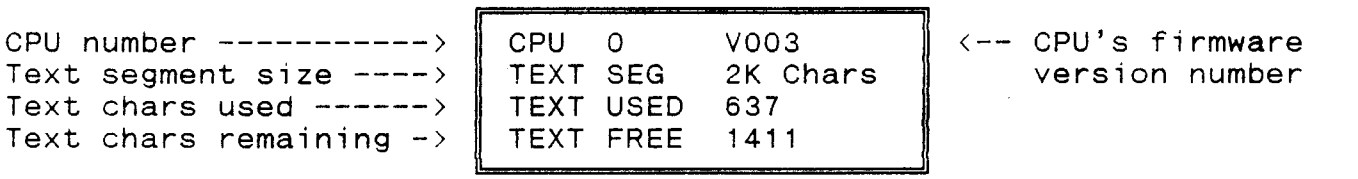
4.5.8 Display Memory map

Displays the memory map, program names and CPU firmware version numbers for all CPUs and the optional LAN2 co-processor. Each display shows information about a single CPU's code or text segment. The map can be paged through using the up and down arrow keys. If connected to a PCD6, the memory map for CPUs 0..6 and the LAN2 can be displayed, the PCD4 has CPUs 0 and 1 or CPU 0 and a LAN2.

Code Segment Display:



Text Segment Display:



Program name — This is the name of the program, taken from the PCD or UPL file which was downloaded. If the CPU doesn't contain a program or the program is not named, this is blank.

Program modified indicator — If a "*" appears after the program name, then the program has been altered using the P1 or P3 ("Write Program" or or "Write teXt"). If modified, the program in the PCD will not match the ".PCD" or ".UPL" disk file with the same name (if present).

Firmware version — The version of firmware (in EPROMs) which controls the CPU. All versions in the same rack must be the same. If the CPU is not present, this is blank.

CODE SEG — This is the allocated code segment size for the CPU, in K lines (1K = 1024 lines = 4096 bytes). NOTE: CPU 0 has 100 lines less code space than the stated code segment size in K.

CODE USED — This is the actual number of program lines used. In the example above, 1959 program lines have been used (0-1958), the next free program line is at address 1959.

CODE FREE — This is the number of program lines which are unused.

TEXT SEG — This is the allocated text segment size for the CPU, in K characters (1K = 1024 characters).

Display Memory map Continued

TEXT USED — This is the actual number of characters used by the existing texts. NOTE: For each text there is an overhead of 3 characters for the text number and the terminating nul (0). In the example above, 637 characters have been used out of a possible 2048. This leaves space for a further 1411 characters (-3 for each text).

TEXT FREE — This is the number of text characters which are unused.

If the CPU has not been allocated a code or text segment, the CODE SEG or TEXT SEG size will be 0.

The memory map must be initialized using the "Write Memory map" command or using P3 Programming Utilities' "Loader" program. Different code and text segment sizes can be allocated for each CPU.

If the CPU does not exist and the memory map has not been initialized, this message is displayed:

```
CPU  2
CODE
     MEMORY NOT
     ALLOCATED
```

If a LAN2 co-processor exists, the last page of the display shows the LAN2 information:

```
Firmware version -->  LAN2
                      VERSION: V003
Station number ---->  STATION: 002
```

These keys control the display:

Up Arrow	Displays the preceding screen.
Down Arrow	Displays the next screen.
Left Arrow	Moves back to the "Display" command menu.
HOME	Returns to the top-level menu.

4.5.9 Display History

Each CPU contains a "History Table" which logs certain failures and undesirable events, and a "Halt Register" which logs the reason for the previous Halt (or the date and time the system was powered up). This command displays the History Table and Halt Register of the connected CPU.

The Halt Register contains a text describing the reason for the last halt. If the CPU has not halted, the message "EVERYTHING IS OK" is displayed (or "MODIFIED PROGRAM" if the program checksum is invalid), and the date and time shows the date and time the CPU was powered up.

The History Table is a circular list of error messages, the date and time of the error, and a location code. It contains up to 16 entries (numbered 0-15) before wrapping around and overwriting older entries. The last entry in the table may be at any position, and is indicated by the text "LAST ENTRY", see below. If the table entry has not been used, "EMPTY" is displayed.

The first display shows the Halt Register (HALT REG). Pressing the up or down arrow from here displays the last (15) or first (0) entry in the History Table. The other History Table entries are displayed by pressing the up or down arrow keys.

Refer to the list in the SBUG Debugger's Functional Specification for details of the Halt Register and History Table messages.

The "location code" is a hexadecimal value which can be used to determine the user program address where the error occurred. Use the formula: adds = (location code - 400190H) / 4. If the result is beyond the end of the CPU's code segment, subtract the size (in program lines) of the preceding CPU's code and texts segments.

Halt Register display example, a Halt instruction has been executed:

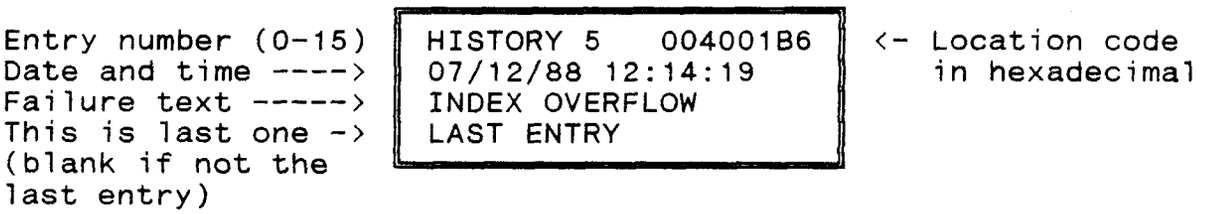
```
Halt Register ----> | HALT REG    004001B6 | <- Location code
Date and time ----> | 07/12/88 12:14:19    |    in hexadecimal
Halt reason ------> | HALT INSTRUCTION     |
                    +----------------------+
```

Halt Register display example, no Halt has occurred:

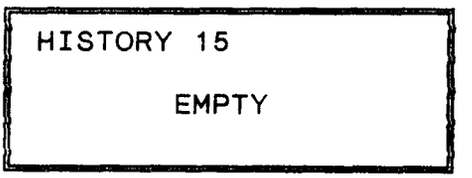
```
                    +----------------------+
                    | HALT REG    00000000 |
Date and time of -> | 07/12/88 12:14:19    |
last power-up       | EVERYTHING IS OK     | <- No halt or errors
                    +----------------------+
```

Display Halt and History Continued

History Table display example, shows the last entry:



History Table example, if the table entry is empty:



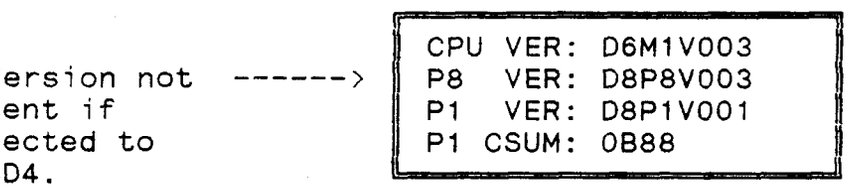
These keys control the display:

Up Arrow	Selects the previous History Table entry.
Down Arrow	Selects the next History Table entry.
Left Arrow	Moves back to the "Display" command menu.
HOME	Returns to the top-level menu.

4.5.10 Display Versions

Displays the firmware versions of the connected CPU, the PCD8.P8 (if used) and the PCD8.P1 itself. Also displays the PCD8.P1's EPROM checksum in hex. If the version contains a '$' symbol, this means it is a pre-release version, which should be upgraded.

The firmware versions of CPUs other than the connected CPU can be seen using the "Display Memory map" command.



4.6 Write Commands

4.6.1 Write Accumulator

Alters the state of the connected CPU's Accumulator (1/0). This command cannot be used if the CPU is in Run or Conditional Run.

4.6.2 Write Status flag

Alters the state of one of the arithmetic status flags (1/0). Each flag, Zero (Z), Negative (N) or Error (E) must be selected separately. To set the Positive (P) flag, set the Negative flag to zero (0). This command cannot be used if the CPU is in Run or Conditional Run.

4.6.3 Write Output and Write Flag

The "Write Output" and the "Write Flag" commands are similar.

These commands have two options. When entering the Output or Flag number at the "<number>" prompt, either a single address or an address range (start-end) can be entered. For example,

Write Flag 0 <ENTER>	Single address given, individual Flags can be manually written.
Write Flag 0-99 1 <ENTER>	Address range given, Flags 0 to 99 are set to 1.

Single address given:

If a single address is entered, then Output or Flag values are displayed as for the "Display Output" or "Display Flag" commands (see section 4.5.2). A cursor can be moved using the arrow keys to select an element to be altered. Units can be selected using SPACE. If hex/bcd units are selected, then hex values 0..F can be entered to write 4 elements at a time. If binary units are selected, the 0 and 1 keys change the value of a single element. New values are written to the PCD immediately the key is pressed. The display is refreshed about once a second, and the + - < > keys work as in the "Display" commands.

Address range given:

If an address range is entered at the "<number>" prompt (start-end, eg. 0-100, 32-63), then the next prompt asks for a value <1, 0>. All Outputs or Flags in the given range are set to this value. For example, the command "Write Output 32-63 1" sets Outputs 32 to 63 (32 Outputs are set high).

Write Output and Write Flag Continued

If an address range is given, writing a large number of elements may take a few seconds. While writing, the "** WAIT **" message is displayed. The command can be aborted using HOME or the right arrow key. To clear all Outputs or Flags, use the "Clear" command, see section 4.7.

Key usage (when manually writing individual elements):

Down Arrow	Moves the cursor down. If on the bottom line, the display is scrolled up, displaying the next 8 elements.
Up Arrow	Moves the cursor up. If on the top line, the display is scrolled down, displaying the previous 8 elements.
Left Arrow Right Arrow	These move the cursor one DIGIT left or right. If on the first or last element on the display, the display is shifted left or right by one digit. If hex/bcd units are selected, the display is shifted by one hex/bcd digit (4 bits).
< >	Scrolls the display left or right by one BIT. This also works when the display units are hex/bcd, but the value of the preceding element becomes the MS (or LS) bit of the first (or last) hex/bcd digit. (The left and right arrow keys shift the display by one DIGIT, not one bit).
0..9, A..F	For entering new values.
SPACE	Selects the display units, binary or hexadecimal/bcd. The hex/bcd displays are the same. Bits are grouped in sets of 4 and displayed as a hex digit (0-F). When hex/bcd units are selected, the keys 0..9, A..F can be used to write new hex/bcd values to 4 elements at a time.
+ -	'+' changes the order of the displayed bits to ascending order, least significant element first. '-' changes the order of the bits to descending order, most significant bit first. This also works when the display units are hex/bcd. This is useful for bcd inputs read using the BITIR or DIGIR instructions.
HOME	Exits and returns to the top-level menu.

4.6.4 Write Register and Write Timer/Counter

The "Write Register" and the "Write Timer/Counter" commands are similar.

These commands, like the "Write Output" and "Write Flag" commands, have two options. When entering the Register or Timer/Counter number at the "<number>" prompt, either a single address or an address range (start-end) can be given.

Single address given:

If a single address is entered, then Register or Timer/Counter values are displayed as for the "Display Register/Timer/Counter" commands (see section 4.5.3). The display is refreshed about once a second. A cursor can be moved up or down to select an element, and a new value can be entered. When the first digit is typed, the existing value is cleared.

Values can be displayed in decimal (bcd), hex, binary, floating point (Registers only) or ASCII (Registers only). The SPACE key selects the display units. Unless the display units are binary, new values can be entered in any units, irrespective of the display units. Values can be displayed in hex and entered in decimal etc.

To enter a hex value, end the number with "H" (eg. ADCFFH). To enter a floating point value, include a decimal point "." or exponent "E" in the value (eg. 1.2, 123E6, -1.234E-6). The enter a binary value end the number with "Q" or "Y" (eg. 1001Q).

The DEL key deletes the digit before the cursor. If all the digits are deleted, the original value is re-displayed.

The new value is not actually written to the PCD until the ENTER key is pressed. If entering binary values from the binary display, ALT-ENTER must be pressed to write the value, individual bits are not written as they are typed.

When the display units are binary, the cursor can be moved to individual bits of the 32-bit value using the arrow keys (the bit number is shown on the display), and new binary digits can be entered. When the first digit is typed, the displayed value stops being refreshed. The new data is not written into the PCD until ALT-ENTER is pressed or another display is selected. If DEL is pressed, the original binary value is restored, and is refreshed.

Address range given:

If an address range is entered at the "<number>" prompt (start-end, eg. 0-100, 10-11), then the next prompt asks for a value. All the Registers or Timers/Counters in the given range are set to this value. The value can be a decimal, hex, binary or floating point value as described above. For example, the command "Write Register 100-199 FFH" sets Registers 100 to 199 (100 Registers) to FF hex (255 decimal).

Write Register and Write Timer/Counter Continued

If an address range is given, to write a large number of elements may take a few seconds. While writing, the "** WAIT **" message is displayed. The command can be aborted using HOME or the right arrow key. To clear all Registers or Timers/Counters, use the "Clear" command, see section 4.7.

*** NOTE ***

Timers and Counters cannot be written with negative or floating point values.

Valid values for Registers:	-2147483648..+2147483647
	0..FFFFFFFFH
	±2.710505E-20..±9.223371E+18
Valid values for Timer/Counters:	0..+2147483647
	0..7FFFFFFFH

Key usage:

Down Arrow	Moves the cursor down to select the next element. If the cursor is on the last line, the display is scrolled up. For the binary display, the next R/T/C is displayed if the cursor was on the last line.
Up Arrow	Moves the cursor up to select the previous element. If the cursor is on the top line, the display is scrolled down. For the binary display, the previous R/T/C is displayed if the cursor was on the top line.
SPACE	Selects the display units, either decimal/bcd (the default), hexadecimal, floating point (Registers only), binary or ASCII (Registers only). Except for the binary display, the display units do NOT affect the units in which data can be entered. ASCII units can be selected, and data can be entered in hex etc.
Left Arrow	Returns to the prompt to enter the R/T/C number. This allows fast selection of another R/T/C without having to press the up/down arrow keys many times. In binary display mode, this key moves the cursor to the next bit on the left (use HOME to exit).
Right Arrow	Works like the down arrow key, unless in binary display mode when it moves the cursor to the next bit on the right (use HOME to exit).
DEL	Deletes the character left of the cursor. If the value is displayed in binary units, DEL restores the original value.
HOME	Returns to the top-level menu.
ENTER	The new value is checked, and if valid is written to the PCD. NOTE: If editing a binary value, ALT-ENTER must be pressed because the right arrow key is used for cursor movement.

4.6.5 Write Program

The program in the connected CPU can be modified with this command. The program is displayed as for "Display Program", except a cursor is shown. The up/down arrow keys move the cursor which selects the program line to be written.

Pressing the first key of a new instruction line clears the existing line from the display and a new instruction or operand can be typed. An existing line can be edited by pressing '>'. This moves the cursor to the end of the line, allowing DEL to delete characters.

Since instructions and operands use both alphabetic (A..Z) and numeric (0..9) characters, a method of selecting the ALT key state without pressing the ALT key has been provided, this allows instructions to be entered with one hand. The ALT key state is initially set so that the alphabetic characters are selected, this is useful when entering a mnemonic. After entering a space, the ALT key state is toggled. The first space selects the numeric characters, if another space is typed the case is toggled again to select the alphabetic characters and so on. To enter the instruction "STH I 0" without using the ALT key, type "S T H _ _ I _ 0" where '_' is a space character. To enter an operand which begins with a digit, type a space first to select the numbers, for example "_ 1 0 0". The ALT key can still be used - and must be used to enter hex values, units postfixes and the 'E' for exponent where spaces are not permitted. An example appears on the page after next.

The use of a space character between the medium type or condition code character and the numeric address is optional.

Operand values can be entered in decimal, hex, floating point (LD instruction only) or binary units. To enter a hex value, end the value with "H" (eg. 1234H). To enter a floating point value, use a decimal point "." or exponent "E" (eg. 1.2, 123E6, -1E-6). To enter a binary value, end the value with "Q" or "Y" (eg. 1001Q). NOTE: No space can appear between the digits and the units specifier (H, Q or Y).

The units for the display of LD, LDH and LDL instructions can be selected using the ALT-SPACE key - decimal, hex and floating point.

To enter absolute line numbers for the JR (jump relative) instruction, precede the address with the "$" character. The relative address will be calculated and displayed. For example, if at address 100, entering "JR L $90" will produce "JR L -10" which jumps to absolute address 90.

Program lines can be inserted and deleted by pressing ALT-INS or ALT-DEL when the cursor is in the first column. This can only be done if the CPU is in Stop, Conditional Stop or Halt. When a program line is inserted or deleted, relative jump instructions are automatically adjusted so that the relative jump addresses still jump to the correct line. If lines are inserted or deleted, a "rEstart" command must be executed before the program can be executed.

While entering a new program line, the DEL (or ALT-DEL) key deletes the character before the cursor. If all characters are deleted with DEL, the original program line is re-displayed. NOTE: The DEL key also works as the "U" character, in cases where both DEL and U are valid, for example when entering the FMUL instruction, the "U" is given priority. In this case, ALT-DEL must be used.

Write Program Continued

ENTER must be pressed to transmit the new line to the PCD. When ENTER is pressed, the instruction line is checked. An error message is displayed if the line is invalid, invalid lines cannot be entered.

The invalid line indicator "?" may be displayed during the entry of multi-line instructions. There should not be any invalid lines once the complete instruction is entered. Use ALT-DEL to delete any unwanted operand lines of an over-written multi-line instruction.

The operands on the line after the LD and COB instructions use TWO program lines. When a 32-bit operand is entered, an extra line is inserted to prevent the next instruction line from being overwritten (unless overwriting an existing 32-bit operand). Note that lines cannot be inserted if the CPU is in Run or Conditional Run.

Be careful if editing the program while the connected CPU is in Run or Conditional Run. If invalid lines are created during the editing of multi-line instructions, this can make the program behave incorrectly.

Function Block (FB) parameters (after the CFB instruction) are NOT checked when entered from the P1. It is possible to enter invalid parameters which will cause the program to fail.

If lines are inserted or deleted, or the block structure is changed, a "Restart" must be done to re-organize the CPU's internal data tables. If a restart is not done, the program may not run correctly.

These keys are valid:

Up Arrow	Moves the cursor up to the preceding instruction line. If on the top line, the display is scrolled down.
Down Arrow	Moves the cursor down to the next instruction line. If on the bottom line, the display is scrolled up.
SPACE	Delimits the mnemonic and operands and also toggles the ALT key state, see next page.
ALT-SPACE	Selects units for the display of LD, LDH and LDL values. For LD (32-bits) the units are decimal (the default), hex or floating point. For LDH/LDL, the units are decimal or hex. (Constants can be entered in decimal, hex, floating point or binary regardless of the units selected for display).
Left Arrow	Returns to the prompt to enter the address. This allows fast selection of another instruction line without having to press the up/down arrow keys many times.
HOME	Returns to the top-level menu.
>	Moves to end of line allowing editing with DEL.
0..9, A..Z, DEL etc.	For entering new instructions and operands.

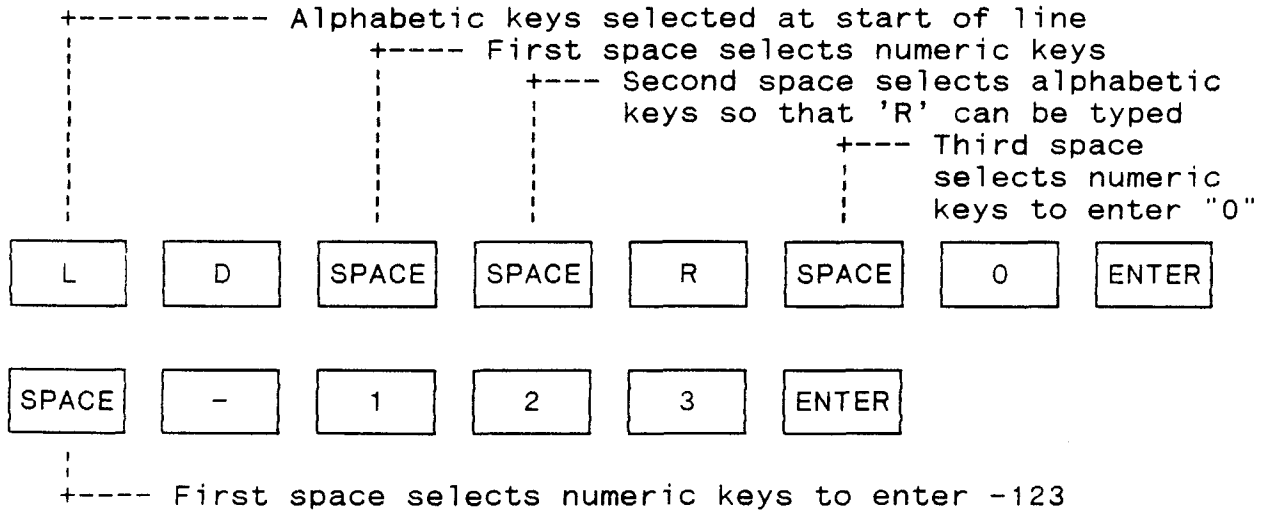
Write Program Continued

$	To enter an absolute address for a JR instruction, precede the address with "$" and the relative address will be calculated an stored, eg. "JR L $10".
ALT-INS ALT-DEL	Inserts or deletes and instruction line at the cursor position (if the cursor is in the first column). This can be done only if the CPU is in Stop, Conditional Stop or Halt. After program lines have been inserted or deleted, a "Restart" must be done before the program can be executed.
DEL	During entry of a program line, this deletes the character in front of the cursor. If all characters are deleted, the original program line is displayed. NOTE: If there is a conflict between DEL and "U", use ALT-DEL.

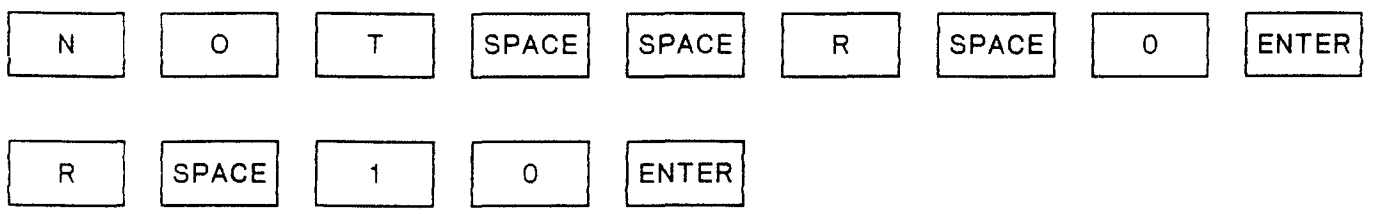
Instruction entry examples

Each SPACE entered toggles the ALT key state. NOTE: The ALT key must still be used to enter hex values, units postfixes and the 'E' for exponent where spaces are not permitted.

To enter the instruction: LD R 0
-123



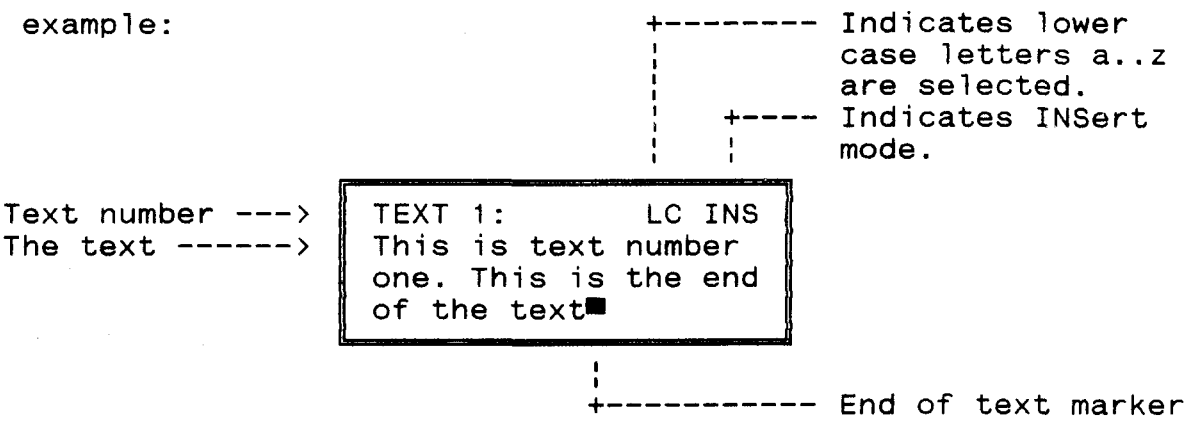
To enter the instruction: NOT R 0
R 10



4.6.6 Write teXt

Texts are displayed as in the "Write teXt" command. A cursor is shown which can be moved using the arrow keys, and text characters can be entered or deleted at the cursor position.

Display example:



Overtype or insert mode can be selected using the INS key, which toggles between the two modes. Insert mode is indicated by "INS" on the top line of the display. If an empty text is written, insert mode is automatically selected.

To enter alphabetic characters, the ALT key must be depressed. Either upper or lower case characters can be entered. Pressing ALT-REP toggles the case of alphabetic characters between upper and lower case. If lower case is selected, "LC" is displayed on the top line.

To enter special characters or characters which are not on the keyboard, enter the decimal ASCII value of the character inside angle brackets <1>..<255>. To put the "<" or ">" characters themselves into a text, these must also be enclosed in angle brackets. The character with ASCII value 0 (NUL) cannot be entered, this is used to terminate the text. If the text is written back to the PCD and then re-displayed, all displayable characters are displayed as the actual character even if originally entered in decimal. ASCII character number examples:

<10>	line feed	<34>	"
<13>	carriage return	<<>	<
<27>	escape	<>>	>

The text is not actually written back to the PCD until ALT-ENTER is pressed. The text is first checked for valid characters inside angle brackets "<..>". If any are invalid (valid range 1..255) or there are any missing "<" or ">" characters, an error message is displayed and the cursor is moved onto the invalid character so that it can be corrected. The size of the text is also checked, it is prohibited to change the size of the text if the CPU is in Run or Conditional Run.

Pressing HOME will abort the text edit if ALT-Y is typed at the "ARE YOU SURE ?" prompt, pressing any other key returns to the edit session.

Write teXt Continued

Key usage:

Arrow keys	These move the cursor around the existing text.
INS	Toggles between insert and overtype mode.
DEL	Deletes the character under the cursor.
ALT-REP	Toggles between upper and lower case mode.
ALT-ENTER	The text is checked, and if valid is written back to the PCD.
HOME	Aborts the text edit after answering ALT-Y to the "ARE YOU SURE ?" prompt, pressing any other key returns to the edit session.
< >	Used to delimit decimal ASCII character values, or the "<" or ">" characters themselves, eg. <<> <>>.
Other keys	Enter the key character into the text.

4.6.7 Write iNdex register

The current Index Register of the connected CPU can be written with a decimal value (0..8191). This command cannot be used if the CPU is in Run or Conditional Run. Note that each COB and XOB has its own Index Register.

4.6.8 Write clocK

The PCD's real time clock is displayed in the same format as the "Display clocK" command, except that a cursor is shown. The display is refreshed until the first digit of the new date and time is typed. All CPUs in the same rack share the same real time clock.

The cursor is moved using the arrow keys, and new digits are entered using the numeric keys. Any digits typed are immediately written into to the PCD's clock.

The HOME key returns to the top-level menu. ALT-ENTER moves back to the previous menu.

NOTE: It is possible to enter an invalid date and/or time. It is the users' responsibility to ensure that the data is correct. For the PCD4, a "COMMAND NOT ACCEPTED" message is displayed if the date or time is invalid.

```
Day of week (1-7) ----> |  DAY:   3          |     Day 1 = Monday
Week of year (1-52) --> |  WEEK:  49         |     Day 7 = Sunday
Date (dd/mm/yy) ------> |  DATE:  07/12/88   |
Time (hh:mm:ss) ------> |  TIME:  15:15:27   |
                        +--------------------+
```

4.6.9 Write passWord

Use this command to temporarily remove the password protection or to change the initial protection level or password. See section 3.7 for protection level details.

This command first prompts for the existing password to be entered. If there is no password, the right arrow key can be pressed without entering the password. Each password character typed is displayed as a "#", this prevents unauthorized personnel seeing the password.

The password can contain any characters A-Z 0-9 @ $ % : ; + - = < >. To select alphabetic characters, the ALT key must be pressed. The password can be up to 8 characters long. Once the password has been typed, press the right arrow key.

```
+-------------------------+
| Write passWord          |
|                         |
| <password> ####_        |
+-------------------------+
```

Write passWord Continued

Two password commands are available:

Remove protection	This sets the password protection level to 0 (no protection). When the P1 is disconnected or the PCD is powered off, the protection level is set back to the defined level.
Change password	This allows a new password to be entered, and the protection level can be changed. See section 3.7 for protection level details.

This is the "Change password" display. The cursor can be moved using the arrow keys, and a new protection level or password can be entered:

```
Change password

Level:    2
Password: HELLO_
```

If either is changed, when the ENTER key is pressed the new level and password must be confirmed:

```
Level:    3
Password: GOODBYE
ARE CHANGES CORRECT?
PRESS ALT-Y IF YES
```

First check that the new protection level and password are correct, then press ALT-Y to store them.

If the protection level or password have been changed by mistake, pressing HOME will abort the changes, first asking for confirmation:

```
Level:    3
Password: HELLO$
DISCARD THE CHANGES?
PRESS ALT-Y IF YES
```

If ALT-Y is pressed, the changed level or password are discarded.

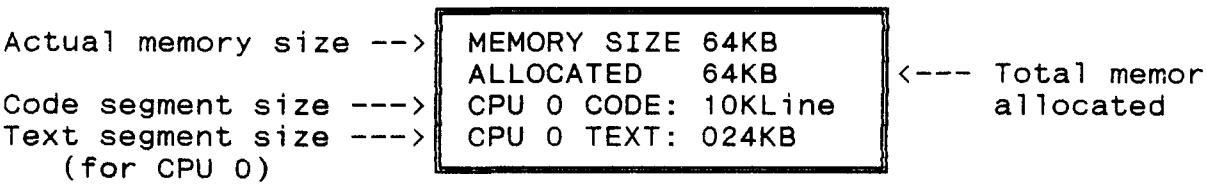
*** NOTES ***

MAKE A NOTE OF THE PASSWORD. The protection level and the password are stored in non-volatile memory within the PCD. If the password is forgotten, the only way to remove the password protection is to remove the battery on the public memory module, or to re-allocate the memory using the PCD Programming Utility's Loader (V1.3 or later).

4.6.10 Write Memory map

The user memory of the PCD is shared by all CPUs in the rack. Using this command, each CPU can be allocated a section of RAM memory according to its needs. Note that ALL CODE AND TEXT IN ALL CPUs is DELETED if the memory is re-allocated. The memory allocation must be done before a new RAM memory module can be used.

Each CPU can be assigned a code segment size and a text segment size, these sizes can be edited on the "Write Memory map" display:

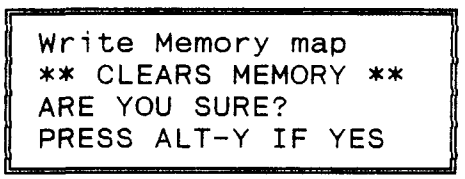


The code segment size is entered in K lines (1K lines = 1024 program lines or 4096 bytes). The text segment size is entered in K bytes (1K bytes = 1024 text characters).

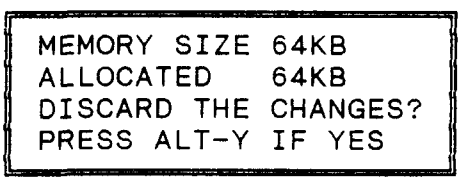
As new segment sizes are entered, the total ammount of memory allocated (in K bytes) is updated on the display. All memory must be allocated so that MEMORY SIZE = ALLOCATED before the new memory map can be stored. All of memory can be allocated to CPU 0 if required (CPU 0, the master CPU, must have a code segment).

The segment sizes for the other possible CPUs (0..1 for the PCD4 and 0..6 for the PCD6) are displayed by pressing the up or down arrows.

To store the new memory map, press ALT-ENTER. The familiar "Are you sure?" prompt is displayed, and a reminder is given that this will clear memory and delete all the existing code and text. Pressing ALT-Y stores the new memory map and clears memory, any other key returns to the "Write Memory map" display:



Pressing HOME will abort, first asking for confirmation if changes have been made:



4.7 Clear Commands

The "Clear" command can clear (set to zero) all Outputs, all Flags, all Timers and Counters, all Registers, or All elements (all the preceding elements). It can also delete the Program and texts or just the teXts.

For safety, the user is prompted "ARE YOU SURE ?" before the clear command is executed. ALT-Y must be pressed to do the clear, depression of any other key will abort. Example of the "ARE YOU SURE ?" prompt:

```
Clear Outputs

ARE YOU SURE ?
PRESS ALT-Y IF YES
```

Program and texts cannot be deleted unless the connected CPU is in Stop, Conditional Stop or Halt mode.

Elements cannot be cleared unless ALL CPUs are in Stop or Halt, because elements are common to all CPUs.

The "Clear" commands are:

```
Clear Outputs
Clear Flags
Clear Timers and Counters
Clear Registers
Clear All elements
Clear Program and texts        } Allowed in Stop, Conditional
Clear teXts                    }  Stop and Halt only
Clear History
```

"Clear Program and texts" and "Clear teXts" do not actually clear any memory. They work by altering the CPU's header to indicate no code or text, then issuing a "Restart Cold" command. When a program is cleared, the first instruction in the code segment is set to NOP.

4.8 Locate Commands

Searches the connected CPU's memory, looking for a Mnemonic, an Operand or an Instruction line.

Locate Mnemonic <mnemonic>

Searches for the instruction mnemonic (ignores the operand).

Locate Operand <[type] value>

This is the most powerful of the "Locate" options. This will find where any element or block is referenced by the operand of an instruction. An optional data type (see below) can be given. If no type is supplied, a constant is assumed. It is not possible to search for condition codes and jump addresses. As described in section 4.6.5, each SPACE entered toggles the state of the ALT key, making one-handed entry of operands easier.

TYPE	DESCRIPTION	TYPE/VALUE EXAMPLE		EXAMPLES OF OPERANDS LOCATED
I	Input	I 0	(*)	STH I 32
O	Output	O 32	(*)	STH I 32
F	Flag	F 100		STH F 100
R	Register	R 10		ADD R 10
T	Timer	T 0	(*)	STH C 0
C	Counter	C 100	(*)	LD C 100
K	K constant	K 10		ADD K 10
TEXT	Text	TEXT 3999		SASI 3999
SEMA	Semaphore	SEMA 99		LOCK 99
=	FB param reference	= 1		STH = 1
COB	Cyclic Org'n Block	COB 0		COB 0, SCOB 0
XOB	Exception Org'n Block	XOB 32		XOB 32
PB	Program Block	PB 200		PB 200, CPB L 200
FB	Function Block	FB 123		FB 123, CFB 123
SB	Sequential Block	SB 10		SB 10, RSB 10
ST	Step	ST 0		ST 0, IST 0
TR	Transition	TR 0		TR 0
Q N B W L D	MOV operands	Q 0, N 1		Q 0, N 1
-	Decimal constant	123		123, 7BH
-	Hex constant	64H		100, 64H
-	Floating point value	1.234		1.234000E+00

* For medium types Input or Output either "I" or "O" can be used. For Timer and Counter either "T" or "C" can be used. For example, locating operand "O 0" will find "I 0" (which may be an Output). Locating "T 0" will find "C 0" (which may be a Timer). References to Inputs and Outputs are both represented by an "I", references to Timers and Counters are both represented by a "C". The P1 does not know the difference Inputs and Outputs or Timers and Counters.

Locate Commands Continued

Locate Instruction <mnemo [mc] [opa]>

Searches for a single instruction line, containing a mnemonic and optional mc/operand. For example: STH I 0, ADD R 100, SASI 3999, NOP. As described in section 4.6.5, each SPACE entered toggles the state of the ALT key, making one-handed entry of operands easier.

Locate From <address>
Locate To <address>

The range of memory to be searched can be set using the "From" and "To" options. Initially, "From" is set to 0, and "To" is set to the last program line.

During the search, the display shows the item being searched for, and the address being examined:

```
Searching for:
STH I 0
From 0 to 6545
Address: 261
```

If the item is not found, the message "*** NOT FOUND ***" is shown on line 3 and the left arrow or HOME key can be used to exit.

If the search is successful, the instruction containing the item is displayed. The instruction display can be scrolled up and down as in "Display Program", see section 4.5.4. To search for the next occurrence of the same item, press ENTER.

The search can be aborted at any time by holding down the HOME or left arrow key for a few moments.

*** NOTE ***

The "To" address is initially set to the last address of the program. If new program lines are entered or inserted using "Write Program", the "To" address is NOT updated until a "rEstart" command is done.

4.9 Instruction Command

This command executes the supplied instruction inside the connected CPU. The CPU must be in Stop, Conditional Stop or Halt.

The instruction can be up to 20 characters long, the ";" character is used to separate each operand. For example:

```
Instruction

<mnemo[mc][opa];...>
ADD  R 0; R 1; R 2
```

If the entered instruction is invalid, an error message is displayed.

As described in section 4.6.5, each SPACE entered toggles the state of the ALT key, making one-handed entry of instructions easier. There may not be room for spaces if a long instruction is entered, in this case the ALT key must be used. It is not necessary for spaces to be used between the medium type and the address, eg. "R0" and "R 0" are both accepted.

Program flow control instructions and definition instructions cannot be executed with the "Instruction" command. These are:

CPB, CFB, EPB, EFB, ECOB, EXOB, PB, FB, COB, XOB,
JR, JPD, JPI,
SB, ESB, CSB, IST, ST, TR, EST, ETR,
RCOB, RSB, NCOB, SCOB, CCOB,
DEFVM, DEFTC, DEFTB, DEFWPR, DEFWPH.

Certain instructions are executed only if the ACCU is high (1). If the ACCU is low (0), the instruction is not executed and an error message is displayed. ACCU dependent instructions are:

COM, SET, RES, SETD, RESD
LD, LDL, INC, DEC are ACCU dependent only if accessing a Timer or Counter.

(+ the indexed versions of these instructions)

5. SELF TESTS

The P1 contains comprehensive self-test facilities. These are designed to aid fault diagnosis during manufacturing and when the P1 is in the field.

The tests are separated into two groups: Power-up tests and a special "Diagnostic" mode.

5.1 Jumpers

After the power-up tests have been successfully completed, the two jumpers JPR1 and JPR2 are checked.

If JPR1 is in, then the keyboard test is entered (see section 5.3). This allows the keyboard test to be selected if the keyboard is faulty.

If JPR2 is in, the power-up tests are repeated until JPR2 is removed. This is to be used for the burn-in tests during manufacturing.

5.2 Power-up Tests

These tests are always run on start-up of the P1, when it is first connected to the PCD or the PCD is powered up.

If any test fails, the P1 attempts to indicate the failure on the display and the P1's microprocessor is halted. To repeat the tests after a failure, the P1 must be powered off and on again.

a) CPU Test

This tests all the microprocessor's internal registers and flags. If this test fails, there will be no indication on the display. The display will not have been blanked.

b) RAM test

Worst-case data retention and addressing test on RAM (random access memory). If the test fails, the RAM chip number (1 or 2) and the address (in hex, 0-7FFF) of the failure are displayed:

```
RAM ERROR

CHIP 1
ADDS 03FE
```

Power-up Tests Continued

c) EPROM Checksum

The EPROM(s) containing the firmware have a checksum, contained in the upper EPROM, which is verified by this test. The checksum is a standard XOR-and-rotate checksum, done on each word in EPROM memory that is used by the program. Unprogrammed words (FFFF's) are not checksummed.

If this test fails, an error message is displayed which shows the actual and the expected checksums in hexadecimal:

```
EPROM CHECKSUM ERROR

ACTUAL      EXPECTED
 03BE         03BB
```

d) Display Test

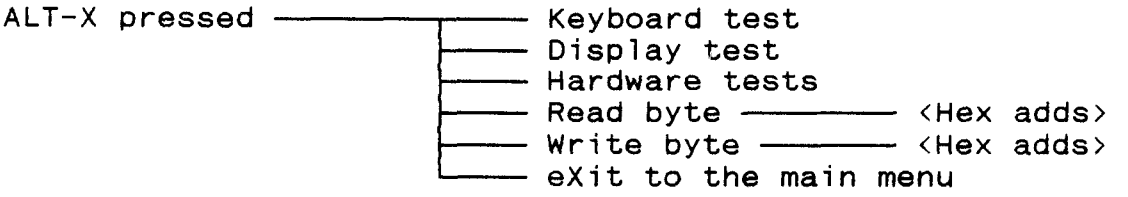
A visual indication of the functionality of each segment of the LCD display is given. This also tests the ability of the microprocessor to write to the display correctly and data retention of the LCD's internal RAM memory.

The display is first blanked, then all segments are set black, then the display is blanked again.

5.3 Diagnostic Mode

To enter Diagnostic mode, press ALT-X from the top-level menu.

The Diagnostic mode menu:



This menu can be stepped through in the usual way with the up and down arrow keys, or the capital command letters can be used. Note that help is NOT available when in Diagnostic mode.

a) Keyboard test

This displays a "map" of the keyboard, with a digit for each key. If the key is not pressed, the digit will be "0", when the key is pressed the digit should be "1". It also shows the character assigned to the last key which was pressed, enclosed in square brackets, eg. [A], [HOME]

The map is organized as two columns of 5 digits. The first row in the first column relates to the first row of keys "ALT" to "D". The last row in the second column relates to the last row of keys "Y" to "ENTER". If any keys do not close, are short-circuited or have faulty diodes, the keyboard map will show it.

If the keyboard is fauly, it may not be possible to enter the keyboard test using the keyboard. In this case, use JPR1 to select the keyboard test on power-up, see section 5.1.

To exit the keyboard test, press ALT-X.

In this example, two keys (ALT-Y) are depressed:

```
MAP:          KEY:
10000 00000   [Y]
00000 00000   (ALT-X
00000 10000    exits)
```

Diagnostic Mode Continued
---

b) Display test

This is a cyclic test which tests the display itself, and the LCD controller's internal RAM. When any character is written into the display RAM, the RAM is read to ensure the character was received correctly. The test runs continuously until it is ended by pressing any key. If a display memory error occurs, the message "DISPLAY ERROR" is displayed, and the test pauses until any key is pressed. The display test consists of these steps:

1) The display is filled with black characters "▒".
2) The display is blanked.
3) The entire character set is displayed, this should also scroll the display up as each line of characters is displayed.
4) The programmable character set RAM is tested by filling it first with horizontal, then with vertical bit patterns. Each bit pattern is read and verified.

c) Hardware tests

Executes the power-up tests continually. This is the same as powering up the P1 with jumper JPR1 in. To exit the tests, the P1 must be powered off and on. The hardware tests are primarily for factory burn-in testing.

d) Read byte

Displays bytes anywhere in the PCD's 68000 microprocessor memory. The hex start address for the display must be given. Four bytes are displayed on a line in hexadecimal. The display can be scrolled up or down using the up or down arrow keys. Press HOME to return to the Diagnostic menu.

```
000018:  00 80 33 74
00001C:  00 80 33 A6
000020:  00 80 33 D8
000024:  00 80 34 0A
```

e) Write byte

This is the same as "Read byte", except a cursor is shown in the data field. The cursor can be moved around using the arrow keys, and new hexadecimal values can be entered. If an attempt is made to alter the data in EPROM (or non-existant memory), the display will not be updated when a hex key is pressed. Press HOME to return to the Diagnostic menu.

f) eXit to main menu

Exits Diagnostic mode and returns to the top-level menu. The P1 can then be operated as normal.

6. ERROR HANDLING

Comprehensive error checking is done by the P1. Errors fall into these categories:

a) Invalid key depressions.

b) Invalid data entered.

c) Illegal commands due to the PCD's operating mode or configuration.

d) Communications errors.

e) Hardware test failures.

f) Internal soft errors.

Error types (a) to (d) are recoverable, the P1 will continue to function. An error message is displayed on line 4 for a few moments.

For error types (e) and (f), the microprocessor in the P1 is halted after displaying a suitable error message (if possible). A power-up reset is required to recover.

6.1 Error Messages

These are grouped in alphabetical order.

** ABORTED **	The operation was aborted, usually because ALT-Y was not pressed as a response to the "ARE YOU SURE?" prompt.
ALL CPUS NOT STOPPED	The operation cannot be done unless ALL CPUs in the same rack are in Stop or Conditional Stop mode.
BAD PCD RESPONSE	The PCD has returned an invalid message which cannot be processed. The command is aborted.
CAN'T WRITE TO EPROM	It is not possible to edit the program or texts if the PCD contains EPROM memory.
CODE SEGMENT FULL	No more program lines can be inserted because program memory is full.
COMMAND NOT ACCEPTED	The PCD could not process the message from the P1 because the command was invalid or there was an error in the PCD. If this occurs when saving a text from the "Write teXt" command, it means that the CPU's text memory is full or has no text segment. The command is aborted.
CPU DOESN'T EXIST	The CPU is not present in the rack.
CPU MUST BE STOPPED	The operation cannot be done unless the connected CPU is in Stop or Conditional Stop.
CPU 0 HAS NO CODE	CPU 0 (the master CPU) must be allocated a code segment using "Write Memory map".
DISABLED BY PASSWORD	The command cannot be executed because the password protection level has not been removed, see sections 3.7 and 4.6.9.
EXTRA OPERAND	Too many operands entered for "Instruction" command.
FATAL INTERNAL ERROR	The P1 detected an internal operating error. This can be caused by a firmware problem or or by a hardware error due to a bad power connection or hostile environment. A text describing the reason or location of the problem is also displayed. Press any key to reset the P1. If the problem persists, notify SAIA Technical Support.
INSTR'N NOT ALLOWED	Program flow control and definition instructions (COB, JR, DEFTB etc) cannot be executed by the "Instruction" command, see section 4.9.
INVALID ADDRESS	The program line number is out of range 0..65535 (PCD6) or 0..16383 (PCD4).

Error Messages Continued

INVALID INSTRUCTION	Not a valid PCD instruction line.
INVALID KEY	The user has pressed a key which in not valid in the present context. Perhaps the ALT key should be used to select the correct key code.
INVALID LINE	The line is not a valid PCD instruction line. Operands of multi-line instructions are checked according to the preceding mnemonic.
INVALID MNEMONIC	Not a valid PCD mnemonic.
INVALID NUMBER	An invalid Input, Output, Flag, Register number or other numeric input. Probably the number entered is too large.
INVALID OPERAND	The operand is not valid or is not complete.
INVALID PASSWORD	The entered password doesn't match the password stored in the PCD, see sections 3.7 and 4.6.9.
INVALID VALUE	The entered number is out of range, or is an invalid hexadecimal or floating point number.
INVALID TEXT	The text to be saved from the "Write teXt" command contains invalid characters in angle brackets, or contains missing or unexpected angle brackets (< >). Valid values in angle brackets are <1>..<255>, <>>, <<>. Values less than 1 or greater than 255 are invalid. The cursor is move to the start of the invalid data.
MAP NOT INITIALIZED	The PCD contains a memory map, which must be first be initialized either by the PCD itself, by the "Write Memory map" command or by the "Up/downloader" program. It is not possible write program lines or texts if the header has not been correctly initialized.
MISSING OPERAND	The instruction needs further operands.
MUST ALLOCATE nnK	All of memory must be allocated before the new memory map can be stored by the "Write Memory map" command.
NO DATA ENTERED	The right arrow cannot be used to select an operation unless data is entered (eg. an address for "Write Program").
NO RESPONSE FROM PCD	The cable connecting the P1 to the PCD is incorrectly wired or is damaged, or the PCD (or P8) is malfunctioning. For the PCD4, this will occur if it has executed a SASI instruction which has re-assigned the PGU serial port - to solve this problem, power up the PCD4 with the P1 connected.

Error Messages Continued

NOT EXECUTED, ACCU=L	The "Instruction" command did not execute the instruction because the ACCU is Low.
NOT IN PRESENT BLOCK	It is not possible to load the instruction pointer with an address that is outside the present block. Occurs with the "Run From" and "Trace From" commands.
NOT START OF INSTRN	It is not possible to load the instruction pointer with an address that is not at the first line of an instruction, it must point to a mnemonic. It is also not possible to set a breakpoint at an address which is not at the first line of an instruction.
NO TEXT DEFINED	The text to be displayed does not exist in the PCD.
NOT VALID IF IN RUN	The command executed cannot be used if the connected CPU is in "Run" mode. The CPU must first be stopped using the "Stop" command.
OUTSIDE CODE AREA	It is not possble to edit the program if the address is outside the allocated code space or beyond the end of the existing program. To add to the program, begin entering new instructions at the last program line.
P8-PCD COMMS ERROR	If connected to the PCD6 via the PCD8.P800 this indicates there is a bad connection between the P800 and the PCD6. This error should not occur if connected to a PCD4.
RESTART REQUIRED	The PCD cannot be put into Run, Conditional Run or Trace until a "rEstart" is done. A "rEstart" must be done after inserting or deleting program lines.
TEXT SIZE CHANGED	The text size was changed via the "Write teXt" command, but the connected CPU is in Run or Conditional Run, the new text cannot be written unless the CPU is in Stop, Conditional Stop or Halt. The "Write teXt" command must be aborted by pressing HOME.

*** END P1.DOC ***

SAIA AG
Industrial Electronics and Components
CH-3280 Murten/Switzerland

Telephone 037 727 111
Telefax 037 714 443
Telex 942 127

Further representatives

Belgique Landis & Gyr Belge SA, Dépt. Industrie
Avenue des Anciens Combattants 190, B-1140 Bruxelles
☎ 02 244 02 11, Tx 65 930, Fax 02 242 88 31

Danmark E. Friis-Mikkelsen A/S
Krogshojvej 51, DK-2880 Bagsvaerd
☎ 045 42 98 63 33, Tx 37 350, Fax 045 42 98 81 40

Deutschland SAIA GmbH
Daimlerstrasse 1 K, D-6072 Dreieich
☎ 06103 8906—0, Fax 06103 89 06 66

España Landis & Gyr BC SA
Batalla del Salado 25, Apartado 575, 28045 Madrid
☎ 91 467 19 00, Tx 22 976, Fax 91 239 44 79

France SAIA Sàrl.
10, Blvd. Louise Michel, F-92230 Gennevilliers
☎ 1 4086 03 45, Tx 613 189, Fax 1 4791 40 13

Great Britain A.S.A.P. Ltd.
Unit 15D, Compton Place, Surrey Avenue, Camberley, Surrey GU 15 3DX
☎ 0276 691 580, Fax 0276 691 581

Italia SAIA S.r.l.
Via Cadamosto 3, 20094 Corsico MI
☎ 02 48600600, Fax 02 48600692

Nederland Landis & Gyr BV, Div. Electrowater
Kampenringweg 45, Postbus 444, NL-2800 AK-Gouda
☎ 01820 65 685, Tx 20 657, Fax 01820 32 437

Norge Malthe Winje & Co A/S
Cort Adelersgt. 14, Postboks 2440, Solli, N-0202 Oslo 2
☎ 02 55 86 40, Tx 19 629, Fax 02 55 22 11

Österreich Landis & Gyr Gesellschaft m.b.H
Breitenfurterstrasse 148, Postfach 9, A-1230 Wien
☎ 0222 80 108-0, Tx 132 706, Fax 0222 00 100 313

Portugal Infocontrol Electronica e Automatismo LDA.
Av. da Igreja No 68-1° Esq., P-1700 Lisboa
☎ 01 7751 61-65, Tx 63 454, Fax 01 7756 87

Suomi
Finnland Landis & Gyr Suomi OY
SF-02430 Masala
☎ 8 029731, Tx 121 039, Fax 8 02975531

Sverige Beving Elektronik AB
St. Eriksgatan 113a, Box 21 104, S-10031 Stockholm
☎ 08 15 17 80, Tx 10 040, Fax 08 33 68 63

USA After sales services; Maxmar Controls Inc.
99 Castleton Street, Pleasantville, New York 10570-3403
☎ 914 747 3540, Fax 914 747 3567

Australia Landis & Gyr (Australia) Pty Ltd
411 Ferntree Gully Road, P.O. Box 202, Mount Waverley, Vic. 3149
☎ 3 544-2322, Tx 32 244, Fax 3 543 7496

Argentina Electromedidor S.A.I.y C.
Defensa 320, RA-1065 Buenos Aires
☎ 1 337125, Tx 23 377, Fax 1 3319582

Subject to change without notice

Printed in Switzerland 26/727 E1 7.90 K01