RCSL No: 99001059

Edition: July, 1988

Author: Bo Bagger

Title:

RC3502-2 Operating Guide

PN: 99200170

A/S REGNECENTRALEN
af 1979

Keywords:

RC3502, OPSYS Commands, DEBUG Commands, Autoloading.

Abstract:

This manual describes how to operate the RC3502 Model 2: Power-on, power-off, commands to the operating system OPSYS, operation of the DEBUG console, autoloading.

(This manual substitutes RCSL No.: 99 0 00771).

(72 printed pages).

Printed by A/S Regnecentralen af 1979, Copenhagen

Users of this manual are cautioned that the specifications contained herein are subject to change by RC at any time without prior notice. RC is not responsible for typographical or arithmetic errors which may appear in this manual and shall not be responsible for any damages caused by reliance on any of the materials presented.

PN: 99200170

TABLE OF CONTENTS PAGE

APPENDICES:

TABLE OF CONTENTS (continued) PAGE

FOREWORD.

This edition of the RC3502-2 Operating Guide is based on components of the following versions:

SW2001,	RC3502-2 Operating System release 3.2 87.03.20
RC3502-2,	DEBUG Firmware ROB584 87.09.23
F505A,	RC3516, RC3517, RC3519 Autoload PROM ROB607,608 version 9.6 88.05.27

Autoload and load from Papertape and GCI are removed.
LAN and X.25/3 autoload are included.

The process and incarnation concepts are changed to the concept of a process as an incarnation of a program. This reflects the language changes in RTP3502 and RTP86.

The changes are marked in the left margin.

June, 1988

1. SWITCHES AND INDICATORS 1.

1.1 Operator´s Control Panel 1.1

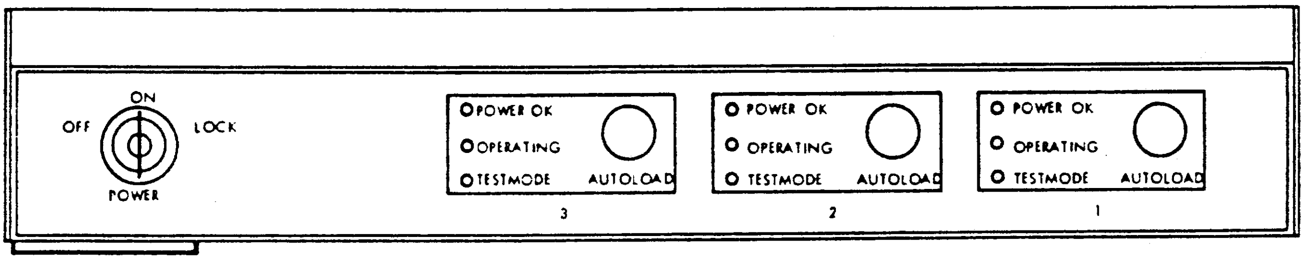


Fig. 1. OCP for Rack with up to Three RC3502

Power off of the RC3502(s) is done by turning the power key to the OFF position.

Power on of the RC3502(s) is done by turning the key to the ON position (or further on to the LOCK position).

The AUTOLOAD button(s) is (are) enabled when the key is in the ON position, and disabled, when in the LOCK position.

The AUTOLOAD button initiates autoloading of the RC3502 in question.

The POWER OK indicator is illuminated during power OK condition on the RC3502.

The OPERATING lamp indicates that the RC3502 is running normally.

The TEST MODE lamp indicates that the RC3502 is executing the built-in test programs.

1.2 Processor Front Panel 1.2

The front panel of the processor boards contain five switches, five indicators, and a jack.

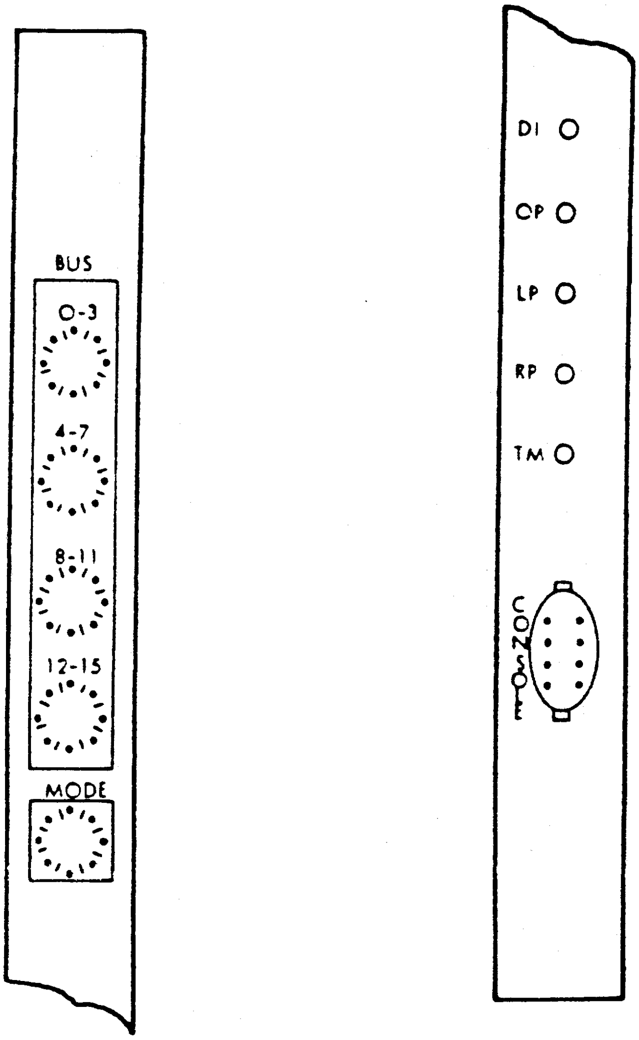


Fig. 2. Processor Front Panel, Switches and Indicators

1.2.1 Switches 1.2.1

All of the switches are rotary switches with 16 positions, indicated by the hexadecimal numbers 0 to F. The switches are set by means of a screwdriver.

1.2.1.1 Bus Switches 1.2.1.

The four switches marked BUS are used to supply the processor with data. There is a switch for bits 0 to 3, 4 to 7, 8 to 11, and 12 to 15.

1.2.1.2 Mode Switch 1.2.1.

The switch marked MODE is used to control the baud rate for the console and the execution of the built-in test programs (sect. 1.2.2).

If the mode switch is equal to, or greater than 8, the console is locked to Terminal-mode (T-mode), i.e. the console will not switch to Debug-mode (D-mode) by activating the BELL key (CTRL G). The mode switch is read continuously, so the baud rate may be changed on a running machine. The current value may be examined by the display command Y3F (see 2.2.2).

Settings		Baud Rate	Execution Mode
0	(8)	300 bps	run test, loop
1	(9)	1200 bps	run test, loop
2	(A)	300 bps	skip test
3	(B)	1200 bps	skip test
4	(C)	300 bps	run test, no loop
5	(D)	1200 bps	run test, no loop
6	(E)	300 bps	skip test
7	(F)	1200 bps	skip test

Test Program Execution Modes

run test The test programs are executed whenever the autoload button is pressed.

skip test The test programs are not executed.

loop	The test programs are executed in an endless loop.
no loop	The test programs are executed once.

1.2.2 Indicators 1.2.2

DI Disabled Interrupt
This lamp, when lit, indicates that the processor is running in the disabled interrupt mode.

OP Operating
This lamp, when lit, indicates that the processor is running normally; when it is extinguished, the processor has stopped.

LP Left Parity Error
This lamp, when lit, indicates that a parity error has been detected during a memory read in the left byte. The lamp can be extinguished only by autoloading.

RP Right Parity Error
This lamp, when lit, indicates that a parity error has been detected during a memory read in the right byte. The lamp can be extinguished only by autoloading.

TM Test Mode
This lamp, when lit, indicates that the processor is executing the built-in test programs. The current program is indicated by the DI, OP, LP, RP, and TM lamps, TM representing the least significant bit of the program number.

If an error is detected by a test program, one of the following messages is displayed on the console. The number in front of the following text is the number shown by the indicators DI through TM:

1 8085 Communication Test

```
Message: ERR00 <dummy><dummy><dummy>
         Y68 gives 6 bytes of transmitted data.
         Y70 gives 6 bytes of received data.
         Y8B is zero after debug/microprogram
             communications timeout.
```

3 Interrupt Test

Message: No message.

Test microprogrammed interrupt of control microprocessor. RP and TM are lit.

5 Working Register Address Test

```
Message: ERR02 <address><errdata><04>
OK data = address.
```

7 Working Register Data Test

```
Message: ERR03 <address><errdata><sub>
sub = 01: if lsb.addr = 0 then OK data = AAAA
                          else OK data = 5555
sub = 02: if lsb.addr = 0 then OK data = 5555
                          else OK data = AAAA
```

7 8085 EPROM Sum Test

```
Message: ERR0B <expected sum><computed sum><dummy>
```

9 Memory Address Test

```
Message: ERR04 <address><errdata><sub>
         Y40 gives the module number under test.
```

The test will read both by means of word and by byte read. In the latter case OK data is the byte contents of the address read by byte read.

```
sub = 02: right parity error = R
      04: left  parity error = L
      06: left and right parity error = LR
      41: dataerror
      43: dataerror + R
      45: dataerror + L
      47: dataerror + LR
```

B Memory Data Test

```
Message: ERR05 <address><errdata><sub>
         Y40 gives the module number under test.
```

```
sub = 02: right parity error = R
      04: left  parity error = L
      06: left and right parity error = LR
      40: dataerror,      okdata = AAAA
                                   in addr 0000
      41: dataerror,      okdata = 5555
      42: dataerror + R,  okdata = AAAA
      43: dataerror + R,  okdata = 5555
      44: dataerror + L,  okdata = AAAA
      45: dataerror + L,  okdata = 5555
```

```
46: dataerror + LR, okdata = AAAA
47: dataerror + LR, okdata = 5555
```

The test will write alternating AAAA,5555.

D <u>Internal Interrupt Test</u>
Message: ERR06 <low,high><errdata><04>

```
low  = byte with lowest  interrupt
high = byte with highest interrupt
```

OK data = high.

F Schedule Test

```
Message: ERR07 <paraml><errdata><sub>

sub = 01: no external interupt
      02: maperror, okdata = 07FF
      03: external interrupt or missing
          "interrupt chain end plug"
      04: maperror, okdata = 0007
      05: coroutine error, okdata = 0007
      06: coroutine error, okdata = paraml
      07: medium priority error,
                            okdata = paraml
      08: low priority error,
                            okdata = paraml
      09: high FF,          okdata = 000F
      0A: medium FF,        okdata = 0017
      0B: low FF,           okdata = 001F
```

11 Interrupt Map Test

```
Message: ERR08 <address><errdata><04>
okdata = address.
```

13 Prefetch Test

```
Message: ERR09 <address><errdata><sub>

sub = 01: load of ICD,      okdata = 5555
      02: load of ICD,      okdata = AAAA
      03: nxtbyte read ICD,okdata = AAAB
      04: nxtword read ICD,okdata = AAAD
      05: nxtbyte read ICD,okdata = address
      06: read of nxtbyte, okdata = address and OFF
      07: ICD,              okdata = address
      08: nxtword even ICD,okdata = address
      09: nxtword,          okdata = address
      0A: odd addr,         okdata = address
```

15 Register Stack Test

```
Message: ERR0A <paraml><errdata><sub>

sub = 01: not stack limit
      02: stack limit
      03: size error, okdata = 0016
      04: stack limit
      05: stack limit
      06: data error, okdata = paraml+7
      08: stack limit
      09: stack limit
      0A: data error, okdata = paraml+7
```

1.3 Power Supply 1.3

The power supply POW207 is supplied with the following controls:

POWER: Circuit breaker, lit when power on.

POWER OK: Indicator which is illuminated during power ok condition.

POWER FAILURE:
OVER-TEMPERATURE:
OVER-VOLTAGE: Error indicators which are illuminated after an error condition. These indicators are reset after activating the circuit-breaker, or after activating the RESET push-button.

RESET: Push-button for manual generation of an autoload signal and a reset of the error indicators. When this control is activated, the voltage to the crate will be turned off.
Use the Autoload button on the OCP to generate an autoload signal without turning off the power.

2. CONSOLE OPERATION 2.

The console may be in one of two possible modes: Debug-mode (D-mode) or Terminal-mode (T-mode). A switch between the two modes takes place when the BELL key (CTRL G) is activated.

2.1 Terminal Mode 2.1

The console may work as terminal for the RC3502 software system, while in T-mode.

An operator process coordinates the communication between the software system and the operator.

2.1.1 Operator Process 2.1.1

Input and output messages to the operator are identified by a name. The operator process holds a variable ´current name´, which identifies the current process for input or output.

´current name´ is updated in the following situations:

1. ESC <name> NL

   The input line contains a name which is assigned to ´current name´. The operator searches its queue of pending input messages for a message with <name> as identification. If at least one message is found, it is activated. If it is a reactivation, the old input is repeated.

   If no message is found, BELL is echoed.

   Note:
   ESC NL is attention to ´current name´.

2. By output.

   If the output message has name = ´current name´, the output message is printed on the console.

   If the output message has name <> ´current name´, ´current name´ is updated and

> "current name"

is printed followed by the text from the output message.

3. ESC ? NL

Prints the identifications of all pending input messages. ´current name´ is updated to the last name in the printout.

The operator has a number of facilities for controlling output and for editing purposes:

CTRL	E	- deletes the whole line
BS		- deletes the last character
ESC NL		- repeats the whole line
RUBOUT		- deletes the last character. ´<-´ is echoed.
CTRL	S	- stops output. May be reset by ESC .
CTRL	Q	- starts output.
CTRL	O	- skips output. The function is alternating between skip and noskip, and is reset to ´no skip´ by ESC

Note:

NL may be Carriage Return or Line Feed.

2.1.2 OPSYS Commands 2.1.2

OPSYS interprets the following commands. The underscored characters are sufficient for the interpretation.

More than one command may be typed on one line, unless the syntax is terminated by <nl>.

Whenever <process> or <program> is listed - unless otherwise stated - we refer to incarnations of two program families declared in ADAM with one of the following headings:

```
PROGRAM pip (VAR sv: system_vector);
PROGRAM pop;
```

In the following all numbers are decimal, unless otherwise stated.

BREAK <process>

The child <process> is broken with the current value of excode as exception code.

The child may be any process in the process tree.

E.g.: BREAK S

CHECK <module>

performs a CRC16 check of the memory module <module>. <module> is the hexadecimal base address. If an error is detected, the word address interval, the expected, and computed checksum are printed.

E.g.: CHECK F2

CREATE <process> $\left\{ \underline{\text{AS}}\ \langle\text{program}\rangle \right\}_{0}^{1}$

creates an incarnation of the program <program>. If AS <program> is omitted the program is supposed to have the same name as the process. The size of the stack is the current value of SIZE (see the SIZE command).

E.g.: CREATE T1 AS TEST CREATE T2 AS TEST

DATE $\left\{ \langle\text{year}\rangle.\langle\text{month}\rangle.\langle\text{day}\rangle\ \langle\text{hour}\rangle.\langle\text{minute}\rangle.\langle\text{seconds}\rangle \right\}_{0}^{1}$

If the parameters are included the date is initialized. The command always responds with the current date.

E.g.: DATE 88.06.14 15.30.20

EXCODE <integer>

initializes the current value of excode to <integer>.

Default: excode = -1.

E.g.: EXCODE 47

FREE $\left\{\text{<module>}\right\}_0^1$ <nl>

lists the free interruption levels and memory bytes. If <module> is specified, the free memory areas in the RAM module <module> are listed. The start displacement and size in bytes of the holes are listed, besides the number of holes, the minimum hole, the maximum hole and the sum of the holes.

FROM FPA

initializes the current value for load kind to FPA. Load kind controls the kind of driver used for dynamic load (see LOAD). Default load kind is taken from the hexadecimal switches.

E.g.: FROM FPA

HELP lists the available OPSYS commands.

IN <inchannel>

initializes the current value of the I/O channel used for load. Default <inchannel> is taken from the hexadecimal switches.

E.g.: IN 81

KILL <process>

works as the REMOVE command.

LINK <program>

links a program with name <program> to a program declaration in ADAM.

E.g.: LINK CPUSE

LIST $\left\{\text{<process>}\right\}_0^n$ <nl>

lists the process tree with root <process>. If <process> is missing ´adam´ is taken as root.
The wildcard character ´*´ may be used in <process> to specify zero or more occurrences of "I don´t care" characters.
If wildcard characters occur, only processes

fulfilling the wild compare operation are listed.
The process state ´fault´ means that the process has entered the ´exit´ state via an exception.
The ´size´ and ´used´ columns denote the size of the process stack and the used part hereof in bytes.

E.g.:

```
LIST *FS*
process       depth  prio  state    size    used   stack   regset  father
fs              1          exit     1000     772  c0.f834    07    adam
bfs             2     -2   wait      764     684  c4.c914    0d    fs
lfs             2     -2   wait     2030     974  c4.c124    0c    fs
fsevent         2     -2   wait      800     306  c4.bcbc    0b    fs
LIST OPSYS
process       depth  prio  state    size    used   stack   regset  father
opsys           0     -1   run       744     642  c0.f28c    06    adam
```

LOAD $\left\{\text{<program>}\right\}_0^n$ <nl>

loads the programs from an external device of kind ´current load kind´ (see the FROM command) in the I/O channel ´current inchannel´ (see the IN command). If no programs are specified, all appearing programs are loaded.

E.g.: FROM FPA IN 81 LOAD PRINTCHAR PRINTNL

A binary relocatable file is loaded from RC8000 via FPA by e.g.:

MAIN35001 = CRC16 $\left\{\text{Bxxxx}\right\}_1^n$

LOG <module>

prints the contents of the MEM205 or MEM206 errorlog, if not empty.

E.g.: LOG C8

$\underline{\text{LOO}}\text{KUP}\left\{\text{<program>}\right\}_0^n\text{<nl>}$

makes a lookup in the LINKER catalog for the listed programs.

If <program> is empty, the whole LINKER catalog is listed. ´LOOKUP PROGRAM´ lists all the programes, ´LOOKUP FUNCTION´ lists all the functions, ´LOOKUP PROCEDURE´ all the procedures, and ´LOOKUP DATA´ all the data programes.
Note: The wildcard character ´*´ may be used in <program> to specify zero or more occurrences of "I don´t care" characters.

E.g.:

```
LOOK *LIN*
>linker
PROGRAM    linker       1988.06.09:09.09  1988.06.09:09.12  4020  3.3
PROCEDURE  fs_link      1986.11.05:02.05  1987.08.25:22.16   272  0.0
PROCEDURE  lf_link      1988.05.16:08.03  1988.05.16:08.10   190  2.14
FUNCTION   link         1988.06.09:08.32  1988.06.09:10.16   224  3.20
FUNCTION   pi3_link     1988.06.09:08.32  1988.06.09:10.16   224  3.20
PROCEDURE  sendlinker   1988.06.09:08.32  1988.06.09:10.16   110  3.20
FUNCTION   unlink       1988.06.09:08.32  1988.06.09:10.16   254  3.20
>opsys
```

The output has the interpretation:

1. program kind
2. program name
3. date of source
4. date of compilation
5. size of object code in bytes
6. version

$$\underline{\text{PRINT}}\left\{\text{<base>}\left\{\text{<first disp>}\left\{\text{<last disp>}\left\{\text{<no of words per line>}\left\{\text{<delta>}\right\}_0^1\right\}_0^1\right\}_0^1\right\}_0^1\right\}_0^1\text{<nl>}$$

outputs the specified memory area with a fixed format (see the example): <base>, <first disp>, and <last disp> are hexadecimal.

<delta> defines the default increase of <first disp> if <last disp> is not specified. After the PRINT command <first disp> := <last disp> + 2.

E.g.:

```
PRINT C0 F28C F2AC
 address
c0.f28c  0000      0    0   0
c0.f28e  8201 -32255  130   1
c0.f290  0000      0    0   0
c0.f292  ffff     -1  255 255
c0.f294  0001      1    0   1
c0.f296  0000      0    0   0
c0.f298  00c0    192    0 192
c0.f29a  8eb5 -29003  142 181
c0.f29c  f573  -2701  245 115     s
c0.f29e  4037  16439   64  55   § 7
c0.f2a0  8064 -32668  128 100     d
c0.f2a2  0003      3    0   3
c0.f2a4  c0f2 -16142  192 242
c0.f2a6  70c0  28864  112 192   p
c0.f2a8  f344  -3260  243  68     D
c0.f2aa  c0f3 -16141  192 243
c0.f2ac  34c1  13505   52 193   4
```

PRIORITY <integer>

initializes the current value for priority used by the START or RUN command.

Default: priority = -2.

E.g.:   PRIORITY -1

REMOVE <process>

removes the child <process> of ADAM and the associated subtree.

E.g.:   REMOVE S

RENAME <old program> <new program>

changes the name of <old program> to <new program>. Only loaded programs (as opposed to autoloaded), which are not accessed by other programs i the system, may be renamed.

E.g.:   RENAME MYPROG XMYPROG

RESTART

removes ADAM and the whole application tree, whereafter a link, create and start of ADAM is executed.

RUN <process> $\left\{ \text{AS <program>} \right\}_{0}^{1}$

links, creates, and starts an process with name <process> of the program <program>.

If <program> is omitted, the program is supposed to have the same name as the process.

E.g.:   RUN T1 AS TEST RUN MIRROR

SIZE <integer>

initializes the current value of SIZE in bytes used when creating ADAM children.

Default: SIZE = 0.

(Note: SIZE = 0 will trigger the use of the default create size for the program).

E.g.: SIZE 1526

START <process>

starts the ADAM child <process> with the current value of PRIORITY as priority.

E.g.: START S

STOP <process>

stops the ADAM child <process>.

E.g.: STOP S

UNLINK <program>

unlinks a program with name <program> from a program declaration in ADAM.

E.g.: UNLINK CPUSE

UNLOAD$\left\{\text{<program>}\right\}_1^n$<nl>

deletes the programs from the LINKER catalog, if the programs are not referenced by other programs. If other programs become not referenced after the delete, these programs are also deleted.

E.g.: UNLOAD CPUSE TEST

2.1.3 Messages from OPSYS 2.1.3

The rest of the command line is skipped if any of the following messages appear:

*** loader not ready

- the LOADER is not included in the system or is unable to run owing to lack of memory.

*** command not implemented

- the command is not available in this version of OPSYS

*** syntax error

- misspelling of a command

*** programname missing

*** unknown process

*** unknown program

*** programname busy

- incarnations of this program still exist

*** processname missing

*** name in use

*** no free programdeclarations

- you must release a program declaration in ADAM by the command UNLINK

*** program not loaded

- the LINKER catalog does not contain the stated program

*** program parameters not equal

- a program with the stated name exists in the LINKER catalog, but the parameter list does not fulfil the declaration

```
PROCESS pip (VAR sv : system_vector);
```

*** size too small or too large

- use the SIZE command to adjust the SIZE parameter

*** program not linked

- use the LINK command

*** unknown program

- the program is not in the LINKER catalog. The program may be loaded by the LOAD command

*** program busy

- the program is still accessed by other programs in the system

2.1.4 Messages from LOADER 2.1.4

*** install more ram memory

- the LOADER cannot get enough memory to run

*** loaddriver no stack

- the driver cannot be created due to lack of memory

*** level reservation trouble

- the interrupt level requested for load is occupied by another program incarnation in the system

scan no: x from fpa in xxxx

- initialize load from RC8000 when the first scan is announced (Note: xxxx is decimal). Later scans are performed by the loader itself with no need for operator assistance.

expected: xxxx
received: xxxx

- the crc16 data check reports an error. The programs should be reloaded

end loader

- normal finis message from the LOADER. A list of the loaded programs is printed with name and compilation date. The list may be extended with the information

*** warning: versionerror

- the program should be recompiled, but loading continues

*** loadfile unintelligible

- the loadfile has the wrong format or is garbage.

*** overlap

- the program is already in the LINKER catalog. The program in the catalog is used instead and the loaded program deleted

*** not defined

- the program was not in the LINKER catalog or amongst the loaded programs.

2.2 Debug Mode 2.2

2.2.1 Activation 2.2.1

If the MODE switch is set in the range 0 to 7, the console may be put in Debug-mode (D-mode) at any time by pressing the BELL key (CTRL and G keys) without stopping instruction execution in the processor. A number of display and control commands become available for technical purposes.

NOTE: In D-mode the commands must be typed with capital letters (e.g. with alpha lock activated).

An illegal command will be displayed as an asterix.

2.2.2 Display Commands 2.2.2

Display commands cause the display of eight words of data. The following display commands are available:

M <addr>
Modify Memory
Displays the contents of the 8 memory words starting at <addr>.

W <regset>
Modify Working Registers
Displays the contents of the 8 working registers comprising registerset <regset>.

P <regset>
Modify Register Stack
Displays the contents of the register stack associated registerset <regset>. At most 8 register stack elements are displayed.

L <level>
Modify Working Registers
Displays the level number, the registerset, and the contents of the 8 working registers comprising the registerset connected to <level>.

Y <yaddr> Modify Control Microprocessor RAM
Displays the contents of the 8 control microprocessor RAM bytes starting at <yaddr>.

Display commands are executed, when a display command is entered.

One can now modify the displayed data by entering new data in the same positions on the following line. Pressing the space bar will move the cursor one position to the right. All updates are byte per byte.

When a P command is terminated (by CR, +, or -) the cursor position defines the number of register stack elements. If the number has been changed, a # is displayed. Note, that a cursor on the first position does not empty the register stack. This is done by the # key (see later).

A display command is terminated by pressing one of the following keys:

#	The # key terminates the P command with an empty register stack. Otherwise # is blind.
CR	The CR key terminates the current display command. The console will await the next command.
+	The + key terminates the current display command and executes a display command for the succeeding 8 words (M), 8 bytes (Y), up to 8 elements (P), or the 8 registers on the succeeding level (W,L).
-	The - key terminates the current display command and executes a display command for the preceding 8 words (M), 8 bytes (Y), up to 8 elements (P), or the 8 registers on the preceding level (W,L).
.	The . key terminates the current display command and executes a display command for the same 8 words (M), 8 bytes (Y), up to 8 elements (P), or the 8 registers on the same level (W,L).

ESC The ESC key terminates the current display command, but no data modification takes place in the M, W, P, and L commands. The text <ESC> is displayed. The console will await the next command.

2.2.3 Control Commands

2.2.3

The following control commands are available:

R Run
The processor will start instruction execution.

S Instruction Step
The processor will execute one instruction, stop, and display the current levelno, the registerset, and the contents of the 8 working registers, and reactivate the console. Modification of the displayed data is not possible. The watchdog timer will stop, if started.

S <steps> Multi-Instruction Step
The processor will execute <steps> instructions, stop and reactivate the console.

Z Instruction Step
The command works as the S command. The P display command is performed implicitly, so the contents of the current register stack is also displayed.

Z <steps> Multi-Instruction Step
The command works as the S <steps> command, but a P display command is performed implicitly for each instruction step.

T <testno> Single Selftest
The processor will execute a single selftest, in a loop mode, according to the following table. If testno is chosen as Cl-D5, then there will be no errormessage, and the test will continue even if an error occurs. The memory test is performed on 64K bytes blocks according to the RAM configuration bit map. See app K.

The test can be terminated by use of the ESC key. Errorno + info are explained in 1.2.2.

A 16 bits pass counter can be used in single test execution. It will be set to zero, when the T command is used, and is incremented by one before each pass. The value may be examined, when the T command has terminated after ESC or after an error. Y65 gives the LSB, and Y66 gives the MSB of the pass counter.

testno without mess	testno with mess	err no	test
00C1	0081	00	fifo test
00C3	0083	-	7.5 interrupt test
00C5	0085	02	W-register address test
00C7	0087	03	W-register data test
00C9	0089	04	memory address test
00CB	008B	05	memory data test
00CD	008D	06	internal intr. test
00CF	008F	07	schedule test
00D1	0091	08	intmap test
00D3	0093	09	prefetch test
00D5	0095	0A	register stack test
00D7	0097	0B	8085 EPROM sum test
00DF	009F	-	´power´ restart, but no reset of controllers

2.2.4 Command Parameters 2.2.4

All numbers entered or displayed are hexadecimal.

At any time the entering of an empty command (i.e. pressing the CR key) will cause the previous command to be repeated.

An address (<addr>) is entered using one of the following formats:

```
     <base> : <disp>
```

or

```
              : <disp>
```

<base> is the leftmost 8 bits of the 24-bit address.

<disp> is the displacement within the selected memory module, i.e. the rightmost 16 bits of the address.

If the second format (: <disp>) is used, the last entered address base will be echoed and used.

3. AUTOLOADING 3.

The autoload function may be initiated by:

- Power Restart
- Watchdog Restart

Power Restart

Power Restart happens:

- when power ON is performed manually on the OCP or on the power supply,
- by temporary power failure,
- by manual activation of the autoload button on the OCP or the AUTO push button on the power supply.

The built-in test programs are activated controlled by the ´MODE´ switch, the CPU initializes the registers, whereafter control is passed to the autoload program residing on the first memory module.

Watchdog Restart

Watchdog Restart may be activated both manually by means of the ´Y´ debug console command and from the software. See app. K.

The CPU initializes the registers, whereafter control is given to the autoload program. No built-in test programs are activated.

3.1 Autoload Switch Format 3.1

The autoload program interprets the four BUS switches on the Processor Front Panel according to the format:

```
  0      1-3             4-7
---------------------------------
! D !   !   !   !   !   !   !   !
! / !  K I N D  ! M O D U L E £ !->
! E !   !   !   !   !   !   !   !
---------------------------------

               8-15
---------------------------------
!   !   !   !   !   !   !   !   !
->!         A D D R E S S           !
!   !   !   !   !   !   !   !   !
---------------------------------
```

D/E — Autoload Disabled/Enabled
is intended for drivers controlling external devices, which may autoload RC3502. A driver may activate the watchdog function if autoload enabled.

0 ~ enabled
1 ~ disabled

KIND — Autoload Kind
defines which algorithm the autoload program executes.

0 LAN + X.25/3
Autoload is from RcLAN. If the autoload fails, an X.25/3 autoload via COM204 is initiated. After autoload the software in EPROMs is included.

1 WD
Autoload is from Winchester Disc, and the software in EPROMs is included.

2 FPA
Autoload is from FPA, and the software in EPROMs is included.

3 WD + COM204
Autoload is from Winchester Disc. If the autoload fails, a COM204 autoload is initiated. After autoload the software in

EPROMs is included.

4 LAN
Autoload is from RcLAN, and the software in EPROMs is included.

5 COM204
Autoload is from COM204 (Intelligent HDLC Controller), and the software in EPROMs is included.

6 EPROM
No autoload from external device. Only software in EPROMs is included.

7 X.25/3
Autoload is X.25/3 via COM204, and the software in EPROMs is included.

MODULE # Module Number
This switch has different interpretations as a function of KIND.

KIND = LAN + X.25/3 (0)
Module# controls the USER name, when the RC3502 tries to LOGON to an FTSSERVER.

KIND	FTSUSER NAME
0	rc3502<ID register>
1-E	rc35021..rc350214
F	rc3502

The ID register postfix is four hexadecimal digits.
The FTSSERVER´s are requested for autoload of the file ´boot3502´ N times in a cyclic way. When this limit is reached, the basic FTSUSER name is changed from rc3502 to rr3502 and the cyclic file requesting is continued N times. If no success, a permanent fall back to X.25/3 autoload is performed.

KIND = WD (1)
The field has the interpretation:

```
      4-7
-----------------
! L !   !   !   !
! U !   C U     !
! N !   !   !   !
-----------------
```

CU - defines the Control Unit number of the Disc Controller on the SCSI bus.

LUN - defines the Logical Unit Number of the Winchester Disc at the Disc Controller.

KIND = WD + COM204 (3)
Interpretation as KIND = COM204 (5).

KIND = LAN (4)
Module# controls the USER name, when the RC3502 tries to LOGON to an FTSSERVER.

KIND	FTSUSER NAME
0	rc3502<ID register>
1-E	rc35021..rc350214
F	rc3502

The ID register postfix is four hexadecimal digits.
The FTSSERVER´s are requested for autoload of the file ´boot3502´ indefinitly in a cyclic way.

KIND = COM204 (5)
The field has the interpretation:

```
      4-7
-----------------
!   !   !   !   !
! F !  F  A  N  !
!   !   !   !   !
-----------------
```

F - defines the type of load request transmitted on the HDLC line:
0 PAXNET loadrequest
1 X.25/3 loadrequest

FAN - defines the maximum relative channel no. used for autoload in a cyclic way if a channel fails. Channels 0,1,...FAN relative to the start channel (see later) are used.

KIND = X.25/3 (7)
Module controls the X.25/3 calling algorithm. If the switch is zero, the possible X.121 addresses are tried in a cyclic way. If the switch differs from zero, the X.121 address with the internal index equal to the switch value is used permanently for repeated calls.

ADDRESS specifies the interruption level (input/output channel) depending on the value of KIND.

KIND	ADDRESS
0	CHANNELx128+OFFSETx16+TABLE
1	SAI201 DATA interruption level
2	FPA100 REC interruption level
3	CHANNELx128+COM204 interruption level
4	MFC20X interruption level
5	CHANNELx128+COM204 interruption level
6	´Not used´
7	CHANNELx128+OFFSETx16+TABLE

When kind is 0 or 7 The field has the interpretation:

```
            8-11           12-15
   ---------------------------------
   ! A !   !   !   !   !   !   !   !
   ! / !O F F S E T!   T A B L E   !
   ! B !   !   !   !   !   !   !   !
   ---------------------------------
```

A/B - 0 indicates A and 1 indicates channel B.

OFFSET - defines the interruption level offset of the load controller from the reference controller (72 decimal).

TABLE - defines the X.121 address table defined in the current version of the RC3502BOOT prom.

Example 0 Autoload from RcLAN via MFC20X with FTSUSER = rc35027 on interruption level 16 (decimal). If the autoload fails after requsting N FTS servers, change FTSUSER to rr35027 and continue with N requests. If the autoload still fails, continue with X.25/3 autoload via COM204, as if the switch settings were 70xx.

```
   --------       --------
   ! 07xx !   or  ! 87xx !
   --------       --------
```

Conventions:

1. MFC20X interruption level must be 16.

Example 1 Autoload from Winchester Disc with LUN = 1 via disc controller with CU = 2, SAI201 data interruption level 83 (decimal) and SAI201 control interruption level 82 (decimal):

```
   --------       --------
   ! 1A53 !   or  ! 9A53 !
   --------       --------
```

Example 2 Autoload from FPA in channel 81 (decimal)

```
--------          --------
! 2x51 !    or    ! Ax51 !
--------          --------
```

Example 3 Autoload from Winchester Disc with LUN = 0 via disc controller with CU = 0, SAI201 data interruption level 87 (decimal) and SAI201 control interruption level 86 (decimal). If the autoload fails, autoload is from COM204 address 92.8000 level 75 (decimal) channel B using PAXNET loadrequest and a fan consisting of 3 channels (see example 5):

```
--------          --------
! 3293 !    or    ! D293 !
--------          --------
```

Conventions:

1. SAI201 data interruption level must be 87.
2. SAI201 control interruption level must be 86.
3. CU of the disc controller must be 0.
4. LUN of the Winchester Disc must be 0.

Example 4 Autoload from RcLAN via MFC20X with FTSUSER = rc3502 on interruption level 16 (decimal).

```
--------          --------
! 4F10 !    or    ! CF10 !
--------          --------
```

Example 5 Autoload from COM204 address 92.8000 level 75 (decimal) channel B using PAXNET loadrequest and a fan consisting of 3 channels:

```
--------          --------
! 52CB !    or    ! D2CB !
--------          --------
```

Conventions:

1. Start address of the reference controller must be 90.0000.

2. Interruption level of the reference controller must be 72.

3. The controllers must be consecutive and increasing memorywise and according to interruption levels (see the following table).

	address	level	channel	
Reference	90.0000	72	A	
Controller			B	
	90.8000	73	A	
			B	
	92.0000	74	A	
			B	
First	92.8000	75	A	
Controller			B	<- 1)
Last	94.0000	76	A	
Controller			B	<- 2)
	94.8000	77	A	
			B	
	96.0000	78	A	
			B	
	96.8000	79	A	
			B	

1) first channel in fan
2) last channel in fan

Example 6 No autoload, only inclusion of software in EPROMs.

```
--------       --------
! 6xxx !   or  ! Exxx !
--------       --------
```

Example 7 X.25/3 autoload via COM204 on interruption level 72 (decimal) channel A using the X.121 address with internal index 2. The X.121 address table is no x.

```
--------          --------
! 720x !    or    ! F20x !
--------          --------
```

Conventions between controller start address and interruption level are identical as for kind 3 and 5.

3.2 Autoload Messages 3.2

```
boot version x.x
```

```
Autoload from mfc level xxH
     user : <FTS User Name>    file : boot3502
```

- FTS User Name is ´rc3502´ appended the contents of the ID Register if the ´module´ switch is zero. ´1´ to ´14´ is appended if the ´module´ switch takes the values 1 to E (hexadecimal). If the ´module´ switch is F nothing is appended.

```
calling : <X.121 address>
```

```
calling : cyclic starting at : <X.121 address>
```

```
** end print
```

- Announcement of autoload is no longer performed. The autoload procedures continue.

```
Autoload from fpa in xxH
```

```
Autoload from com204 addr xx.xxxx level  xxH  channel
xxH
```

```
Autoload from eprom
```

```
Autoload from wd in  xxH (D) xxH (C) CU xx lun xx
     file : /autoloadcat/<entryname>
```

- <entryname> is taken from the Control Microprocessor RAM. If the first character in this name is zero, the default name ´boot3502´ appended the CU of the RC3502 is used.

*** undefined switchkind xxH

....

- a full stop is printed for every program loaded.

:::

- a colon is printed for every RAM module, which contains programs with a checksum catalog.

;;;

- a semicolon is printed for every PROM module, which contains programs with a checksum catalog.

boot *** exception: xxxx at: xx.xxxx

- consult appendix I for interpretation of the exception code. The autoload is restarted after one minute.

*** level not installed : xx

- the requested interruption level is not installed. The autoload program is restarted.

*** install more memory at : xx

- more RAM memory must be installed to hold the autoloaded programs. The autoload is restarted.

**** warning: versionerror at xx.xxxx

- the program identified by the address must be recompiled to be autoloaded. Consult the output from CROSSLINK for identification. The autoload is restarted.

**** warning: sumerror module : xx

- the crc16 data check reports an error in the EPROM module xx. Operation continues.

**** warning: <program> not included

- the basic program <program> should be included in the autoloaded programs or in EPROMs. Operation continues.

*** sumerror module : xx

- the crc16 data check reports an error in the RAM module xx. The autoload program will start all over again.

last-block xxxxH
new-block xxxxH

- block sequence error. The autoload program will start all over again.

*** harderror : xx

- a harderror occurred during autoload from Winchester Disc. The autoload program is restarted.

02 SCSI bus phase error in the selection phase.

09 No connection to the SAI201.

0a Timeout of interrupt from SAI201 control channel in the selection phase.

0b Hard error on SAI201. Reset of SAI201 failed.

12 Parity error on the SCSI bus in the selection phase.

22 SCSI bus phase error in the command phase.

2a Timeout of interrupt from SAI201 control channel in the command phase.

32 Parity error on the SCSI bus in the command phase.

42 SCSI bus phase error in the data input phase.

4a Timeout of interrupt from SAI201 control channel in the data input phase.

52 Parity error on the SCSI bus in the data input phase.

62 SCSI bus phase error in the data output phase.

6a Timeout of interrupt from SAI201 control channel in the data output phase.

72 Parity error on the SCSI bus in the data output phase.

82 SCSI bus phase error in the status phase.

8a Timeout of interrupt from SAI201 control channel in the status phase.

92 Parity error on the SCSI bus in the status phase.

a2 SCSI bus phase error in the message phase.

aa Timeout of interrupt from SAI201 control channel in the message phase.

b2 Parity error on the SCSI bus in the message phase.

*** filelimit : root xxxx

- The catalog with name ´autoloadcat´ is not found in the root catalog. The autoload program is restarted.

*** filelimit : autoloadcat xxxx

- The announced autoloadfile is not found in the catalog ´autoloadcat´. The autoload program is restarted.

*** filelimmit : <autoloadfile> xxxx

- The announced autoloadfile is unintelligible. The autoload program is restarted.

3.3 Generating Autoload Files 3.3

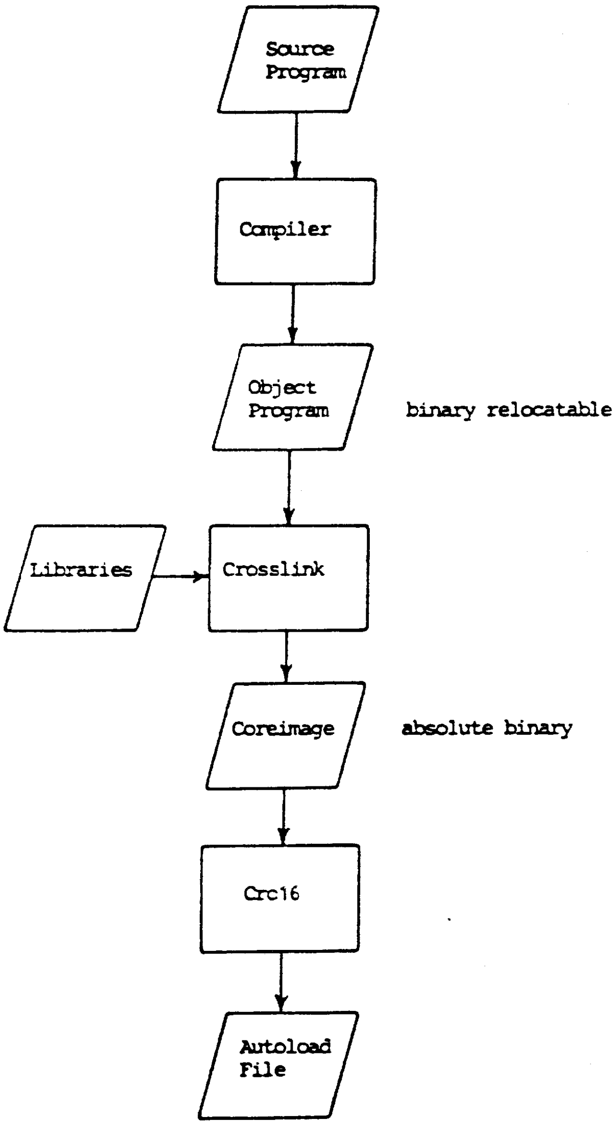


Fig. 3. Autoload File Generation

3.3.1 Generating an FTS Autoloadfile 3.3.1

If the file ´coreimage´ has been produced on RC8000 by the ´CROSSLINK´ program, the following call will generate a file with a format for autoload via RcLAN by the FTS protocol. The file ´boot3502´ must be placed at the FTS Server under the proper catalog bases or directory. The FTS Username is controlled by the ´module´ switch on RC3502.

```
boot3502 = convertplib coreimage
```

3.3.2 Generating an FPA Autoload 3.3.2

If the file ´coreimage´ has been produced on RC8000 by the ´CROSSLINK´ program, the following call will autoload the RC3502 if connected via the process ´main35001´:

```
main35001 = crcl6 coreimage
```

3.3.3 Generating TES202 Eproms 3.3.3

Consult ref. 6.

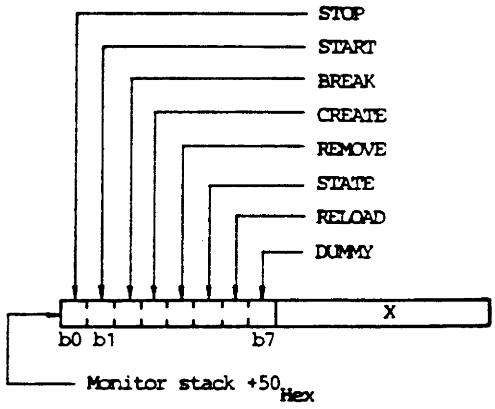
4. ERROR PROCEDURES 4.

For use in error situations it might be useful to get testoutput from the autoload program ´BOOT´ or from the program MONITOR which starts and stops program incarnations.

4.1 Monitor Testoutput 4.1

The monitor stack address is obtained by the LIST MONITOR command to OPSYS or from the testoutput from BOOT (see 4.2).

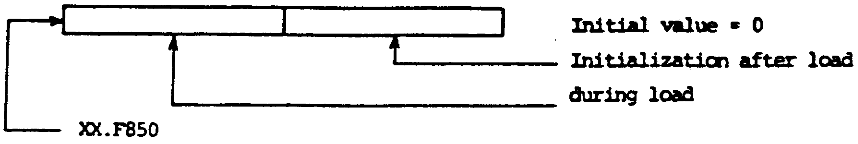
If a bit is set in the byte, the testoutput associated the bit will be generated.



Initial value = 0.

4.2 Boot Testoutput 4.2

The BOOT stack address is xx.F800 where xx is the base address of the last RAM memory module in the RC3502. If testoutput during autoload from FPA is wanted, you must increase the software time out in the FPA xmt driver in RC8000 from 1 sec. to at least 5 sec.



A. REFERENCES

1. PN: 99000484
   PASCAL80, Report

2. PN: 99104398
   PASCAL80, User´s Guide

3. PN: 99103509
   PASCAL80, Driver Conventions

4. RCSL No. 52-AA1192
   RC3502-2 Reference Manual

5. PN: 99000994
   RC3502-2 Real-Time Pascal
   Reference Manual

6. PN: 99103221
   RC3502 PROM Blasting Program
   User´s Guide

7. RCSL No. 52-AA1177
   CPU212-219 Technical Manual

8. RCSL No. 52-AA1197
   RC3502-2 Microprogram Listing

9. PN: 99000990
   Debugger Listing ROB984

B. OPSYS COMMANDS B.

```
break <process>

check <module>

create <process>  as <program>

date  <year>.<month>.<day> <hour>.<minute>.<seconds>

excode <integer>

free <module> <nl>

from fpa

help

in <inchannel>

kill <process>

link <program>

list <process> <nl>

load <program> <nl>

log <module>

lookup <program> <nl>

print <base> <firstdisp> <lastdisp>
       <no_of_words_per_line> <delta>          <nl>
```

```
priority <integer>

remove <process>

rename <oldprogram> <newprogram>

restart

run <process>  as <program>

size <integer>

start <process>

stop <process>

unlink <program>

unload <program> <nl>
```

C. AUTOLOAD SWITCH LAYOUT C.

```
              0      1-3            4-7
            ---------------------------------
            ! D !   !   !   !   !   !   !   !
            ! / !  K I N D  ! M O D U L E £ !->
            ! E !   !   !   !   !   !   !   !
            ---------------------------------

                              8-15
            ---------------------------------
            !   !   !   !   !   !   !   !   !
          ->!         A D D R E S S         !
            !   !   !   !   !   !   !   !   !
            ---------------------------------
```

D/E - Autoload Disabled/Enabled

disabled = 1
enabled = 0

Intended for drivers controlling External Devices, which may autoload RC3502. The drivers may activate the watchdog function, if D/E = 0. BOOT ignores this switch.

KIND

0 LAN + X.25/3
1 WD
2 FPA
3 WD + COM204
4 LAN
5 COM204
6 EPROM
7 X.25/3

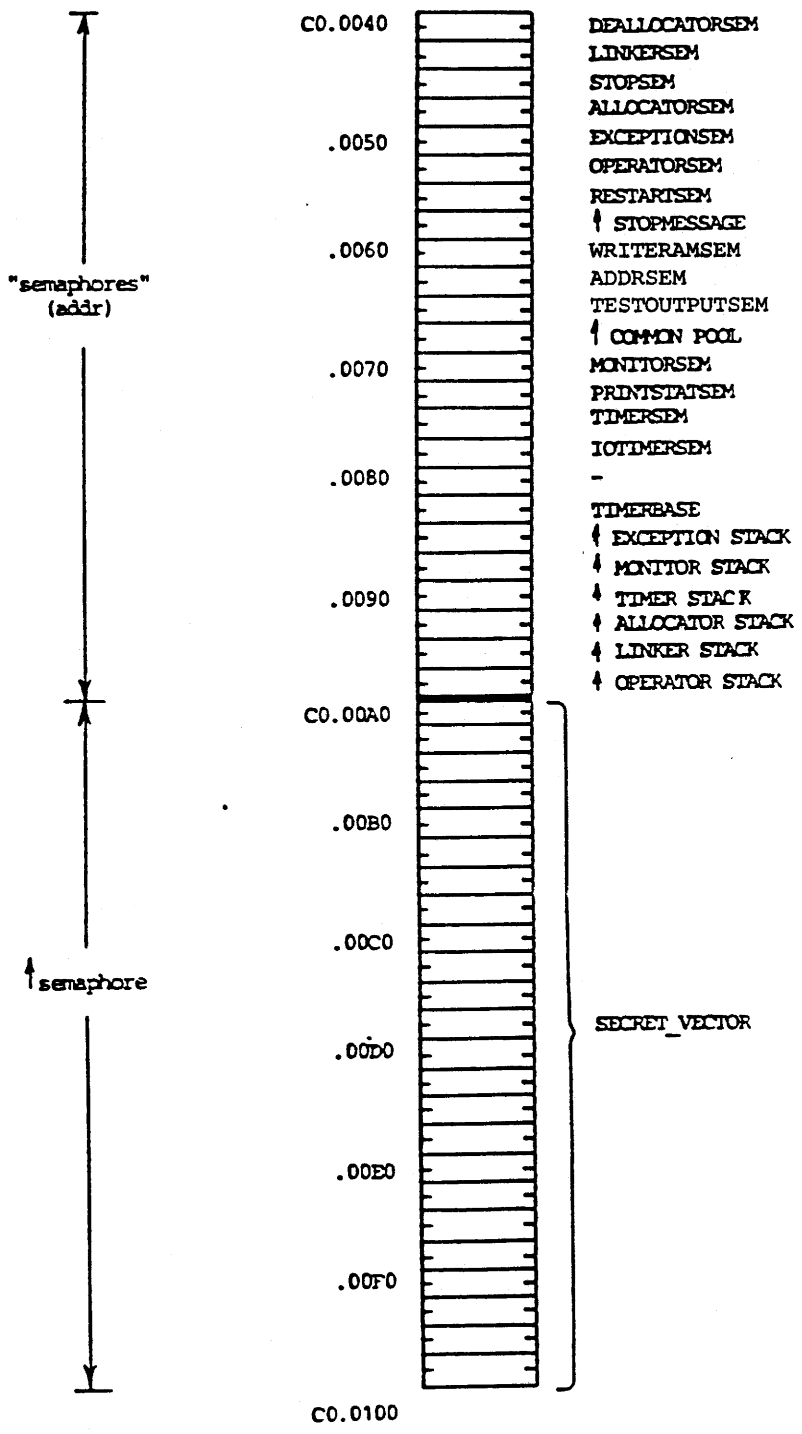
ADDRESS

SAI201 DATA interruption level (CONTROL = DATA -1)	if WD
Interruption level	if FPA or LAN
Channelx128+interruption level	if COM204 or WD+COM204
Channelx128+offsetx16+table	if X.25/3 or LAN+X.25/3

MODULE#

LUN x 8 + CU	if WD
Load request x 8 + fan	if COM204 or WD+COM204
Cyclic or specific call	if X.25/3
FTS User Name	if LAN or LAN+X.25/3

D. SECRET VECTOR LAYOUT D.



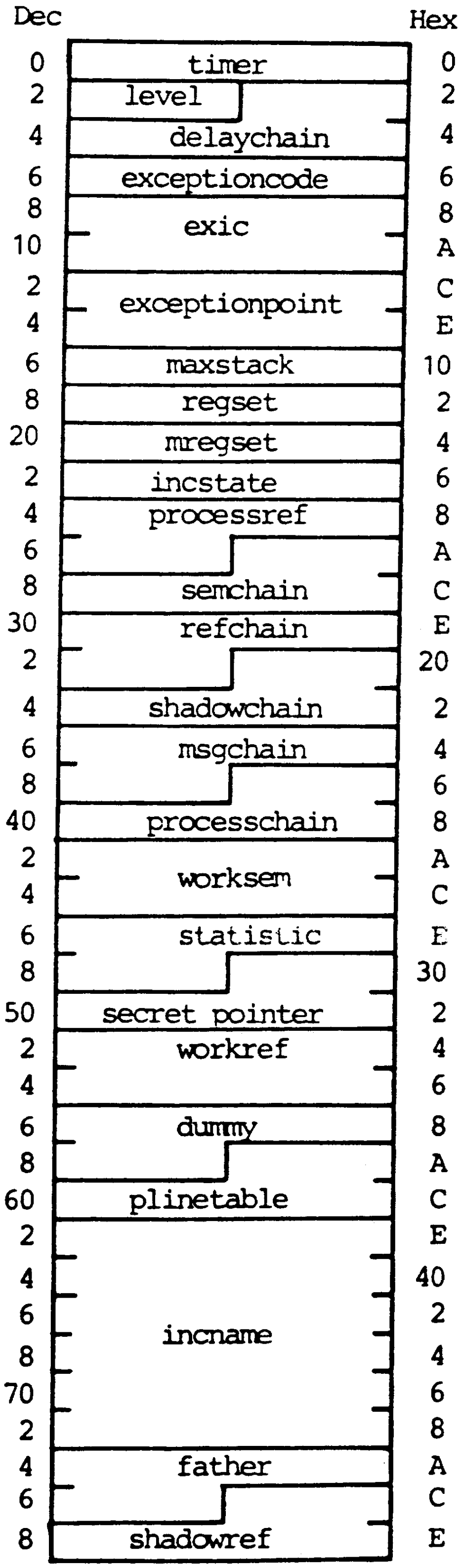
E. INSTRUCTION CODES E.

```
00: except          2b: iofbx           55: not             80: mwt             ab: revlws    1     d6: stvla     2
01: rec1            2c: mtrh      1 1   56: setcr           81: mrecha          ac: moveb           d7: revla     2
02: rec2            2d: iorsc           57: settm           82: svsb2           ad: indh02    2     d8: stvsf
03: rec3            2e: mtrs            58: setatm          83: rvsb2           ae: movea           d9: revsf
04: rec4            2f: mnoop           59: seteq           84: svsb4           af: revpw           da: rvlbns    1
05: rec5            30: uadhwl          5a: setsb           85: rvsb4           b0: readw           db: rvlwns    1
06: rec6            30: reasdl *        5b: setsp           86: svsb6           b1: crout           dc: rvlans    1
07: rec7            31: ult             5c: setun           87: rvsb6           b2: stvaw     2     dd: rvldns    1
08: rec8            32: eq              5d: setin           88: rec0            b3: revaw     2     de: crram
09: rec9            33: ne              5e: setdi           89: revgbs    1     b4: reaod     2     df: cwram
0a: rec10           34: lt              5f: setad           8a: stvlbs    1     b5: reaisd    1     e0: mwist
0b: rec11           35: gt              60: mwis            8b: revlbs    1     b6: stvlw     2     e1: mcist
0c: rec12           36: le              61: mcis            8c: stnhb     2     b7: revlw     2     e2: reaid     1
0d: rec13           37: ge              62: jmzeq     2     8d: indh01    2     b8: revsh     2     e3: rvsd2
0e: rec14           38: indh0     2 2   63: jmzne     2     8e: renpb           b9: rvsh1           e4: reald     2
0f: rec15           39: indh1     2 2   64: jmzlt     2     8f: renhb     2     ba: revsw     2     e5: rvsd4
10: csign           3a: topen           65: jmzgt     2     90: readb           bb: rvsh3           e6: uadhw     2
11: crele     1     3b: tlock           66: jmzle     2     91: crget           bc: revsa     2     e6: reasd   * 2
12: movebs          3c: teqad           67: jmzge     2     92: stvgb     2     bd: rvsh5           e7: rvsd6
13: cinwa           3d: tnill           68: jmprw     2     93: revgb     2     be: revsd     2     e8: rechd     2 2
14: coutwa          3e: mmul            69: jmphc     3     94: setre           bf: rvsh7           e9: revods    1
15: csens           3f: madd            6a: jmpga     2     95: resta     1     c0: mwst            ea: stvlds    1
16: lrele           40: cwait           6b: jmcht     3     96: stvlb     2     c1: inth0     2     eb: revlds    1
17: mtime           41: msub            6c: intrs           97: revlb     2     c2: svsa2           ec: setst
18: mhalt           42: uadd            6d: index           98: stvsb     2     c3: rvsa2           ed: indh12    2
19: except          43: usub            6e: intpa           99: svsb1           c4: svsa4           ee: stcea
1a: csell           44: add             6f: inpdv           9a: stvsw     2     c5: rvsa4           ef: revpd
1b: cstop           45: sub             70: iorbbc          9b: svsb3           c6: svsa6           f0: rvsh0
1c: cstart          46: umul            71: iorbb           9c: stvsa     2     c7: rvsa6           f1: svsh0
1d: iocci           47: udiv            72: iowbbc          9d: svsb5           c8: rechws    1     f2: rvsw0
1e: cslev           48: umod            73: iowbb           9e: stvsd     2     c9: revoas    1     f3: svsw0
1f: cgreg           49: mul             74: iorbwc          9f: svsb7           ca: stvlas    1     f4: rvsa0
20: mwi             4a: div             75: iorbw           a0: mwit            cb: revlas    1     f5: svsa0
21: iowc            4b: mod             76: iowbwc          a1: mcit            cc: revsm           f6: rvsd0
22: iors            4c: and             77: iowbw           a2: svsw2           cd: indh11    2     f7: revld     2
23: iorw            4d: or              78: pcals0    3     a3: rvsw2           ce: reaods    1     f8: stvld     2
24: ioww            4e: xor             79: pcals1    3     a4: svsw4           cf: realds    1     f9: revod     2
25: ioao            4f: crc16           7a: pcals     3     a5: rvsw4           d0: mbset     2     fa: stvod     2
26: iogi            50: neg             7b: pexit     2     a6: svsw6           d1: mbtes     2     fb: reaad     3
27: ionci           51: abs             7c: lpush           a7: rvsw6           d2: stvaa     2     fc: reard     2
28: mrest           52: compl           7d: lpop            a8: rechw     2     d3: revaa     2     fd: reaxd
29: mstst           53: shc             7e: lrese           a9: revgws    1     d4: ststc           fe: cexch
2a: iocda           54: shc8            7f: llock     2     aa: stvlws    1     d5: restc           ff: except
```

F. INSTRUCTION MNEMONICS

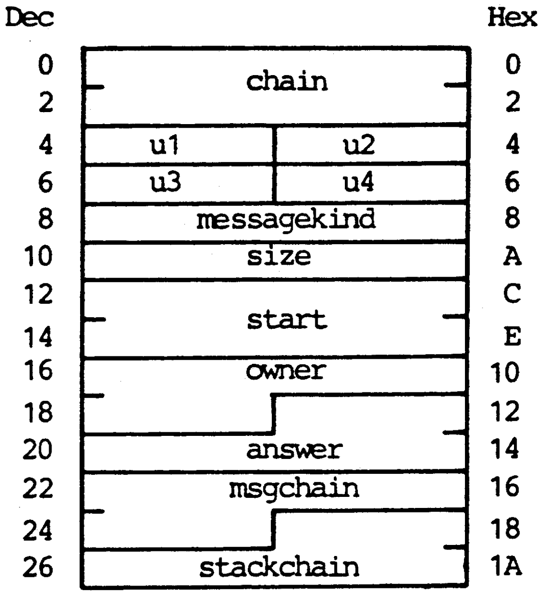
abs	51	ioibx	2b	movebs	12	rec4	04	rvsa6	c7	stvla	d6
add	44	ionci	27	moveg	ae	rec5	05	revsb	b8	stvlas	ca
and	4c	iorbb	71	mrecha	81	rec6	06	rvsb0	f0	stvlb	96
cexch	fe	iorbbc	70	mrest	28	rec7	07	rvsb1	b9	stvlbs	8a
cgreg	1f	iorbw	75	mstst	29	rec8	08	rvsb2	83	stvld	f8
cinwq	13	iorbwc	74	msub	41	rec9	09	rvsb3	bb	stvlds	ea
compl	52	iors	22	mtime	17	rec10	0a	rvsb4	85	stvlw	b6
coutwq	14	iorsc	2d	mtrh	2c	rec11	0b	rvsb5	bd	stvlws	aa
crc16	4f	iorw	23	mtrs	2e	rec12	0c	rvsb6	87	stvsa	9c
crele	11	iowbb	73	mul	49	rec13	0d	rvsb7	bf	svsa0	f5
crget	91	iowbbc	72	mwi	20	rec14	0e	revsd	be	svsa2	c2
crput	b1	iowbw	77	mwis	60	rec15	0f	rvsd0	f6	svsa4	c4
crram	de	iowbwc	76	mwist	e0	rechws	c8	rvsd2	e3	svsa6	c6
csell	1a	iowc	21	mwit	a0	renhb	8f	rvsd4	e5	stvsb	98
csens	15	ioww	24	mwst	c0	renpb	8e	rvsd6	e7	svsb0	f1
csign	10	jmcht	6b	mwt	80	resta	95	revsf	d9	svsb1	99
cslev	1e	jmpga	6a	ne	33	restc	d5	revsm	cc	svsb2	82
cstart	1c	jmphc	69	neg	50	revga	d3	revsw	ba	svsb3	9b
cstop	1b	jmprw	68	not	55	revgas	c9	rvsw0	f2	svsb4	84
cwait	40	jmzeq	62	or	4d	revgb	93	rvsw2	a3	svsb5	9d
cwram	df	jmzge	67	pcals	7a	revgbs	89	rvsw4	a5	svsb6	86
div	4a	jmzgt	65	pcals0	78	revgd	f9	rvsw6	a7	svsb7	9f
eq	32	jmzle	66	pcals1	79	revgds	e9	setad	5f	stvsd	9e
exception	00	jmzlt	64	pexit	7b	revgw	b3	setatm	58	stvsf	d8
exception	19	jmzne	63	reaad	fb	revgws	a9	setcr	56	stvsw	9a
exception	ff	le	36	readb	90	revla	d7	setdi	5e	svsw0	f3
ge	37	llock	7f	readw	b0	rvlans	dc	seteq	59	svsw2	a2
gt	35	lpop	7d	reagd	b4	revlas	cb	setin	5d	svsw4	a4
index	6d	lpush	7c	reagds	ce	revlb	97	setre	94	svsw6	a6
indh0	38	lrele	16	reaid	e2	rvlbns	da	setsb	5a	sub	45
indh01	8d	lrese	7e	reaisd	b5	revlbs	8b	setsp	5b	teqad	3c
indh02	ad	lt	34	reald	e4	revld	f7	setst	ec	tlock	3b
indh1	39	madd	3f	realds	cf	rvldns	dd	settm	57	tnill	3d
indh11	cd	mbset	d0	reard	fc	revlds	eb	setun	5c	topen	3a
indh12	ed	mbtes	d1	reasd	e6	revlw	b7	shc	53	uadd	42
inpdv	6f	mcis	61	reasd1	30	rvlwns	db	shc8	54	uadhw	e6
inth0	c1	mcist	e1	reaxd	fd	revlws	ab	stcea	ee	uadhw1	38
intpa	6e	mcit	a1	rechd	e8	revpd	ef	stnhh	8c	udiv	47
intrs	6c	mhalt	18	rechw	a8	revpw	af	ststc	d4	ult	31
iocci	1d	mmul	3e	rec0	88	revsa	bc	stvga	d2	umod	48
iocda	2a	mnoop	2f	rec1	01	rvsa0	f4	stvgb	92	umul	46
iogi	26	mod	4b	rec2	02	rvsa2	c3	stvgd	fa	usub	43
iogo	25	moveb	ac	rec3	03	rvsa4	c5	stvgw	b2	xor	4e

G. PROCESS DESCRIPTOR LAYOUT G.



Dec		Hex
0	timer	0
2	level	2
4	delaychain	4
6	exceptioncode	6
8	exic	8
10		A
2	exceptionpoint	C
4		E
6	maxstack	10
8	regset	2
20	mregset	4
2	incstate	6
4	processref	8
6		A
8	semchain	C
30	refchain	E
2		20
4	shadowchain	2
6	msgchain	4
8		6
40	processchain	8
2	worksem	A
4		C
6	statistic	E
8		30
50	secret pointer	2
2	workref	4
4		6
6	dummy	8
8		A
60	plinetable	C
2	incname	E
4		40
6		2
8		4
70		6
2		8
4	father	A
6		C
8	shadowref	E

H. MESSAGE HEADER LAYOUT H.



I. EXCEPTION CODES I.

Code (Hex)	Meaning
1	- parity error
2	- registerstack error
3	- undefined opcode
4	- odd number of bytes
5	- stack overflow
6	- pointer = nil
7	- signal: reference = nil
	- push: first param = nil
	- pop: second param = nil
	- lock: reference = nil
	- reference = nil
8	- wait: reference <> nil
	- pop: first param <> nil
9	- push: param locked
	- pop: second param locked
	- signal: reference locked
	- reference locked
A	- lock overflow
B	- arithmetic overflow
C	- index out of bounds
	- subrange out of bounds
D	- illegal zonestate
E	- field overflow
F	- move wraparound
10	- push: identical arguments
11	- push: first param not empty
12	- lock: size error
	- size too small
13	- top <= offset
14	- lock: not data message
	- not data message
15	- not channel message
16	- word block i/o: odd number of bytes
17	- block i/o at level 0
18	- setcr: first limit negative
19	- setad: truncation error
1A	- no resources
1B	- file does not exist
1C	- position outside file
1D	- wrong answer
1E	- setpriority: illegal priority
1F	- pool: no core
20	- process = nil
21	- arithmetic overflow
22	- system error
23	- system error

24	- illegal switch in case construction
25	- upper limit in call of succ
26	- lower limit in call of pred
27	- with: size error
28	- lockdata: top < computed top
29	- local reference variable not nil at routine exit
2A	- local process variable not nil at routine exit
2B-2E	- system error
2F	- break by father

J. WORKING REGISTER LAYOUT J.

Description of the working registers:
(for a more detailed description, refer to the reference manual, ref. 4)

W:	Index:	
00	000-007	regset 0
01	008-00F	regset 1
-		-
79	3C8-3CF	regset 121
7A	3D0-3D7	work regset for multiplications
7B	3D8-3DF	masks0
7C	3E0-3E7	masks1
7D	3E8-3EF	breakpointset
7E	3F0-3F7	monitorset
7F	3F8-3FF	com8085

regset 0 thru regset 121:

1st reg.	ps
2nd reg.	sb
3rd reg.	gf
4th reg.	lf
5th reg.	lu
6th reg.	lm
7th reg.	ib
8th reg.	ic

breakpointset:

1st reg.	breakpointmode (8000 means breakpoint active)
2nd reg.	breakpointbase
3rd reg.	breakpointdisp
4th reg.	unused
5th reg.	unused
6th reg.	unused
7th reg.	unused
8th reg.	unused

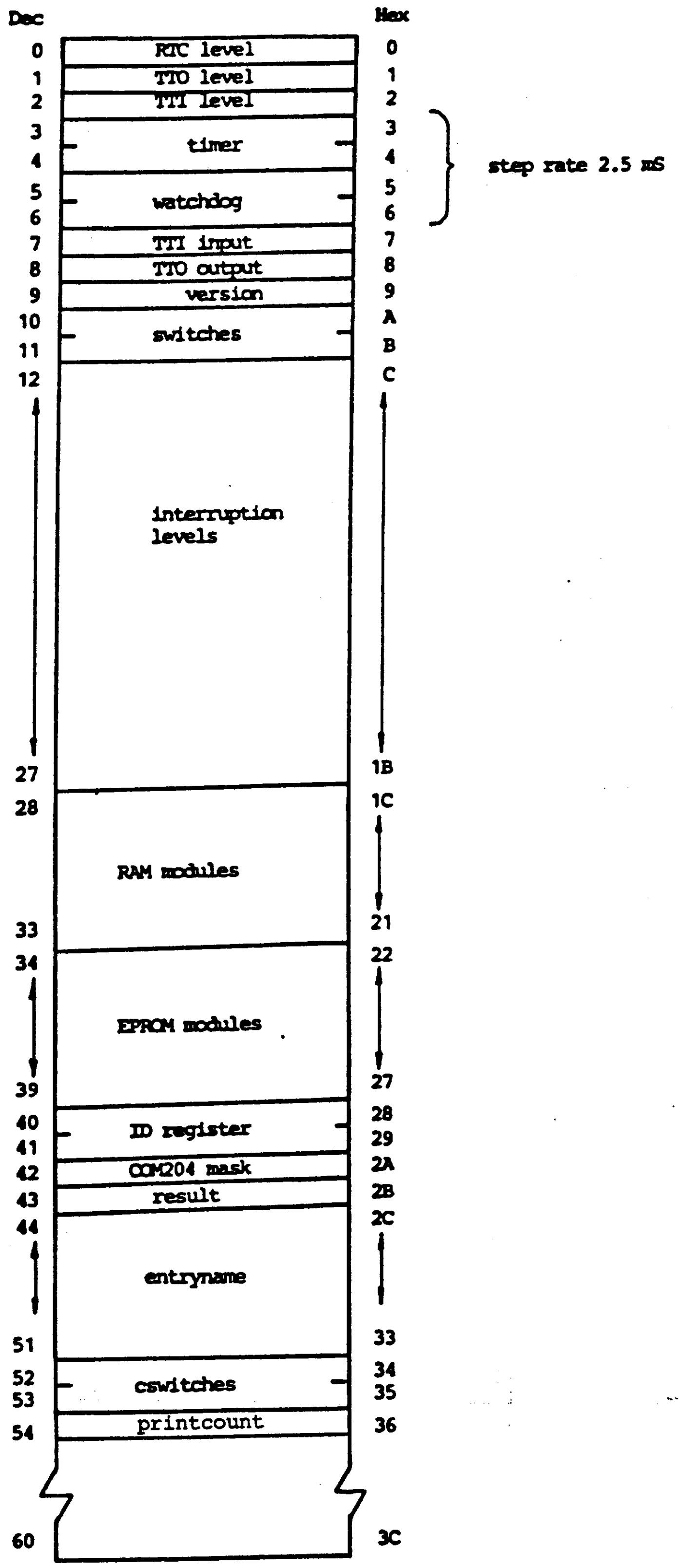
monitorset:

1st reg.	memregsetbase
2nd reg.	memregsetdisp
3rd reg.	waitqueuelast
4th reg.	waitqueuefirst
5th reg.	monitorlevel
6th reg.	unused
7th reg.	unused
8th reg.	unused

com8085:

1st reg.	parityerrorregset
2nd reg.	fifo0, fifo1
3rd reg.	fifo2, fifo3
4th reg.	fifo4, fifo5
5th reg.	cow (value, disp)
6th reg.	message.errorcode
7th reg.	parityerrorbase
8th reg.	parityerrordisp

K. CONTROL MICROPROCESSOR RAM LAYOUT K.



Dec		Hex
0	RTC level	0
1	TTO level	1
2	TTI level	2
3 4	timer	3 4
5 6	watchdog	5 6
7	TTI input	7
8	TTO output	8
9	version	9
10 11	switches	A B
12 27	interruption levels	C 1B
28 33	RAM modules	1C 21
34 39	EPROM modules	22 27
40 41	ID register	28 29
42	COM204 mask	2A
43	result	2B
44 51	entryname	2C 33
52 53	cswitches	34 35
54	printcount	36
60		3C

step rate 2.5 mS

RTC level:
Real Time Clock interruption level.

TTO level:
Console output interruption level.

TTI level:
Console input interruption level

Timer:

```
---------------
! Low ! High !
--------------
   3     4
```

RC3502 is interrupted on level ´RTC level´ every (256 x High + Low) x 2.5 mS.

Watchdog:

```
--------------
! Low ! High !
--------------
   5     6
```

Count down is performed every 2.5 mS. RC3502 is reset, when (if) watchdog.high is decremented from 1 to 0.

TTI input:
When an input character from the console is ready, the character is delivered here and RC3502 interrupted on level ´TTI level´.

TTO output:
RC3502 delivers an output character for the console here, and is interrupted on level ´TTO output´, when the character is printed.

Version:

```
-----------------
! Model ! Micro !
-----------------
        9
```

Model is 1 for RC3502-1 and 2 for RC3502-2. Micro is the micro program version number.

Switches:

```
          10           11
 ----------------------------
 ! 0-3 ! 4-7 ! 8-11 ! 12-15 !
 ----------------------------
           A            B
```

Contains the current value of the four switches marked BUS on the Processor Front Panel.

Interruption Levels:

Describes the 128 interruption level configuration. Level i exists if bit no i is set in the Interruption Level bitmask. Levels 0-7 are described in byte 12, Levels 8-15 in byte 13, etc.

RAM Modules:

Is a bitmask, describing the RAM module configuration. A module is 64Kb. A module exists, if the corresponding bit is set in the bitmap.

Modules A0, A2, ..., AE are described in byte 28, modules B0, B2, ..., BE in byte 29, etc.

EPROM Modules:

Is a bitmask, describing the EPROM module configuration. A module is at most 64Kb. A module exists, if the corresponding bit is set in the bitmap. Modules A0, A2, ..., AE are described in byte 34, modules B0, B2, ..., BE in byte 35, etc.

ID register:

```
      40    41
 --------------
 ! 0-7 ! 8-15 !
 --------------
      28    29
```

The contents of the IDR201 register may be examined here.

COM204 mask:
Controls the COM204 channels during down line load. If bit i is set the channel is skipped, otherwise a load request frame is transmitted on the channel.

Result:
If a down line load is successful, this byte contains the resultant relative channel plus 8, where 0 <= channel <= 7.

If autoload is from Winchester Disc, the CU address of the RC3502 on the SCSI bus is passed here.

Entryname:
Controls autoload from Winchester Disc.

If the fist character in Entryname is zero, the default name ´boot3502x´ is used, where x is the CU address of the RC3502 on the SCSI bus.

Otherwise Entryname is used for lookup in the catalog ´Autoloadcat´.

Cswitches:
Contains a copy of the field switches.

The contents of switches is transferred to Cswitches, whenever the autoloadprogram is started, and the first byte of Cswitches equals zero. The contents of Cswitches controls the autoload program.

Printcount:
Controls the number of times autoload announcements is performed.

L. INSTALLATION STANDARDS AND RECOMMENDATIONS L.

This is a recommendation concerning installation of hardware modules in the RC3502. The guidelines concern interruption levels, input/output priorities, module number selections, DMA priority.

L.1 Input/Output Modules L.1

The priorities and interruption levels in the following table should be followed according to input/output channels and priority. The first module in a group should be the lowest channel number.

L.2 Memory Modules L.

Memory modules are strapped to cover one or more module addresses in the memory address space. They must not overlap. There must exist at least one RAM memory module with address C0 and at least one PROM memory module containing the BOOT program in module address E0. If the autoloaded programs occupy more than one module, there must be RAM memory modules with address C2, C4, etc. MEM204 RAM memory modules can be placed in the address space C0-DE. MEM205, MEM206, MEM207, and MEM209 RAM memory modules can use 80-FE. A special action can be taken on MEM204 to disable the PROM area, which is always equal to RAM module address + 20 (hex). The PROM memory module address on MEM205 and MEM206 may be set completely free in the whole address space, or may be disabled. The PROM areas E0 or E2 may be disabled on MEM207 and MEM209.

TES201 and TES202 can be strapped in the address space C0-FE. (Note, that C0 must always be used as RAM memory).

L.3 Controllers

COM204 occupies 2 positions. The one position connects to the Back Plane Bus. In the second position a CBL714 must be placed.

By convention the first COM204 must be strapped to dual port memory module address 90 (hex) and interruption level 72 (decimal). See page 29.

IMS208 through IMS212 occupies one position.

By convention the first IMS2XX must be strapped to dual port memory module address 80 (hex) and interruption level 8. The succeeding IMS2XX controllers must be set to 80.4000 and level 9, i.e. memory module addresses increase by 4000 (hex) and interruption levels by 1.

L.4 Summary

```
    ---------------------------------------------
    !    !    !    !    !    !    !    !    !    !
    !    !    !    ! 2. ! 1. ! 2. ! 1. ! 2. ! 1. !
    !    !    !    !    !    !    !    !    !    !
    ! C  ! C  ! C  ! I  ! I  ! C  ! C  ! V  ! V  !
    ! P  ! P  ! P  ! O  ! O  ! O  ! O  ! C  ! C  !
    ! U  ! U  ! U  ! M  ! M  ! M  ! M  ! O  ! O  !->
    ! 2  ! 2  ! 2  ! 2  ! 2  ! 2  ! 2  ! 2  ! 2  !
    ! 1  ! 1  ! 1  ! 0  ! 0  ! 0  ! 0  ! 0  ! 0  !
    ! 5  ! 6  ! 7  ! 2  ! 2  ! 4  ! 4  ! 1  ! 1  !
    !    !    !    !    !    !    !    !    !    !
    ---------------------------------------------

    --------------------------------------------------
    !    !    !    !    !    !    !    !    !    !    !
    ! 2. ! 1. ! 2. ! 1. ! 2. ! 1. ! 2. ! 1. ! 2. ! 1. !
    !    !    !    !    !    !    !    !    !    !    !
    ! C  ! C  ! M  ! M  ! I  ! I  ! M  ! M  ! T  ! T  !
    ! O  ! O  ! F  ! F  ! M  ! M  ! E  ! E  ! E  ! E  !
  ->! M  ! M  ! C  ! C  ! S  ! S  ! M  ! M  ! S  ! S  !
    ! 2  ! 2  ! 2  ! 2  ! 2  ! 2  ! 2  ! 2  ! 2  ! 2  !
    ! 0  ! 0  ! 0  ! 0  ! X  ! X  ! 0  ! 0  ! 0  ! 0  !
    ! 5  ! 5  ! X  ! X  ! X  ! X  ! X  ! X  ! X  ! X  !
    !    !    !    !    !    !    !    !    !    !    !
    -------------------------------------------------------
```

The following three points should be observed:

1. The interruption level priority chain starts at the CPU and must not be broken by empty positions until the last module, which uses interruption level priority. At the end of the chain, there must be an "interrupt chain end plug" (CBL735).

2. The DMA priority chain starts at the CPU and must not be broken by empty positions until the last module, which uses DMA priority.

3. The module, which is closest to the CPU, has highest priority, both according to interrupt level and DMA.

Module name	Interrupt level (Hex)	Dual Port Memory Base (Hex)	No. of interrupt levels (Decimal)	No. x salesnumber
	Lowest priority			
CPU	00-04	-	5	
SPARE	05-07	-	3	
IMS2XX	08-0F	80-8E	8	8 x RC3542
MFC20X	10-1F	-	16	16 x RC3532
COM205	20-2F	-	16	16 x RC3547
VCO201	30-37	-	8	8 x RC3583
COM204	48-4F	90-96	8	8 x RC3546
IOM202	50-6F	-	32	4 x RC3521
SPARE	78-7E	-	4	
	Highest priority			

M. INDICES M.

M.1 Survey of Figures M.1

M.2 Catchword Index M.2

RETURN LETTER

Title: RC3502-2 Operating Guide **RCSL No.:** 99001059

A/S Regnecentralen af 1979/RC Computer A/S maintains a continual effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback, your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability, and readability:

Do you find errors in this manual? If so, specify by page.

How can this manual be improved?

Other comments?

Name: ______________________ Title: ______________________

Company: ______________________

Address: ______________________

Date: __________

Thank you

. . . . Fold here . . . .

. . . . Do not tear - Fold here and staple . . . .

Affix
postage
here

A/S REGNECENTRALEN
af 1979

Information Department
Lautrupbjerg 1
DK-2750 Ballerup
Denmark