RCSL: 31-A29
Author: H. Kold Mikkelsen
Edited: July 1971

RC 4000

TEST OF PERIPHERAL DEVICES

OPERATOR's MANUAL

KEY: RC 4000, Test of Peripheral Devices, Operator's Manual

A/S REGNECENTRALEN
Falkoneralle 1
DK 2000 Copenhagen F

CONTENTS:

Preface:

This manual is an substitution of RCSL: 51-VB473 by Jørgen Lindballe.
All test programs designed for the previous loader will work in loader 2.

LOADER, PROCEDURES AND INTERRUPTION

The loader 2 (which reads and stores procedures and testprograms) and 9 procedures (used by the programs for output on and input from the operator's typewriter, for output on a specified output device, for reservation of buffer area and for administration of the test) exist in the binary version on one paper tape.

This paper tape is read from the RC 2000 Paper Tape Reader (dev. No. 0) when the operator activates the RESET and then the AUTOLOAD push-button.

The loader first writes:

channel no. of operator key =
dev. no. of operator's typewriter =

and the operator types the interrupt channel No. of the operator key and the device No. of the typewriter he wants to use. (These numbers may be altered later by activating the RESET and then the START push-button after which the loader writes the above-mentioned questions again).

The loader now writes:

loader 2
input from dev.no.:

and then the operator writes the number of the device, from which the binary testprograms shall be read, followed by <NL>. Consequently he must write 0, if input is wanted from the paper tape reader. If input is wanted from a magnetic tape station (dev. No. > 0), the loader asks:

file no.

and the operator writes the number of the file in which the testprograms are placed. These filenumbers appear from this table:

RC NO.	KIND OF PERIPHERAL DEVICE	FILE NO.
2000	Paper Tape Reader	
150	Paper Tape Punch	
315	Typewriter	
610	Lineprinter, Data Products	
333	Lineprinter, Anelex	
4191	Incremental Plotter	
707	Magnetic Tape Station, 7 tracks	
709	Magnetic Tape Station, 9 tracks	
4415	Drum	
4314	Disc	
	Interval Timer	
	Teletypewriter	
	Display	

When all the testprograms (not more than 15) for the kind of device are stored, the loader writes:

```
<name of the kind of peripheral device>
device no. =
```

After this the operator types the device number (or the device numbers, as it is possible to write up to 3 device numbers separated by <comma>; e.g. teletypewriter). When the program hereafter writes:

```
channel no. =
```

the operator types the interrupt channel number of the device (or the channel numbers, as it is possible to write up to 3 channel numbers separated by <comma>; e.g. teletypewriter).
Now the program asks for an output device:

```
output dev. no. =
```

and the operator types an integer >0.
Hint: programs which uses the output dev. may work as move programs by specifying the cpu-timer (dev. 3) as output device.

Next the program writes:

```
testprogram:
```

and the operator may type a or b or c or ... and in this way select the first, the second, the third, ... testprogram. If he types <NL> the following directory of the stored testprograms is written:

```
a    <description of     1st testprogram>
b    <description of     2nd testprogram>
c    <description of     3rd testprogram>
.                        .
.                        .
.                        .
     <description of the last testprogram>
testprogram:
```

Having selected the testprogram the operator to the question:

number of runs =

must write the number of times the program is wanted to be executed. This number must be chosen so that

$1 \leq$ number of runs $\leq 8\ 388\ 607$

Before the execution of some runs it writes:

run no. <run no.>

namely before the execution of

1st, 2nd, ..., 9th run, if $1 \leq$ No. of runs ≤ 9
1st, 11th, 21st, ..., 91st run, if $10 \leq$ No. of runs ≤ 99
1st, 101st, 201st, ..., 901st run, if $100 \leq$ No. of runs ≤ 999
etc.

Having executed the specified number of runs, the program writes:

test end

and now it is possible to select a new testprogram.

Using the loader some erroneous situations may occur:

If 'end of tape' appears in the paper tape reader or if 'tape mark' appears on magnetic tape when the loader reads testprograms before the first testprogram has been stored, it writes:

mount paper tape

end of file

respectively. If the operator types <NL> after the first message, the loader continues to read.

The error messages:

parity error in <program description>

and (when loading from the paper tape reader):

checksum error in <program description>

means parity error and checksum error in the program being loaded.

If bit 0, 2, 3, 4, 5 or 6 is set in the statusword when the programs are loaded from tape, the loader writes:

status = <bit 0-9>

and the programs must be loaded again.

It is always possible to break the execution of a testprogram by activating the operator key. After this the interrupt sequence writes:

select

Then the operator may type t, o, d, l or c after which the typewriter continues to write the below-mentioned underlined texts:

testprogram:

The operator may select a new testprogram for the same device number.

output dev. no. =

A new output dev. number may be selected.

device no. =

The operator may select a new device number (and after this as usual a new interrupt channel number) for the same kind of peripheral device.

loader 2

Now a new set of testprograms may be loaded, and in this way a new kind of peripheral device may be selected.

core store contents

The contents of some part of the core store may be written on the output device. When the program writes:

first word addr. =

last word addr. =

the operator specifies the part of the core store; it must be mentioned that for the testprograms which use input-output buffer, the addresses of the first and the last bufferword are written on the typewriter immediately before the execution of the first run.

When the program writes:

mode =

the operator types t, d, b or i after which the typewriter continues to write the below-mentioned underlined texts:

text

The bitpatterns are written as a text.

decimal

The bitpatterns are interpreted as integers (negative, zero or positive) and written in decimal.
The program waits for input from the console used for the print layout:

```
        <answer> ::= <integer> ! <empty>
where                   0 <= <integer> <= 24 and
                        <empty> ::= <integer> = 24
```

For <integer> = 0 the program converts the integer to 24.
If <integer> <0 or <integer>>24 the program asks for a new input.
Now the binary word is divided into a number of blocks from the left side, containing the specified number of bits and is printed as separate decimal numbers. If the division 24/<integer> not comes right, the rest word is printed as a binary number, e.g.

decimal 10

will cause the leftmost 20 bits to be separated into blocks of 10 bits and the rest of 4 bits is printed as a binary number.

binary

The bitpatterns are interpreted as positive integers and they are written in binary.
The program waits for input from the console:

```
<answer> ::= <integer> ! <empty>

where

0<= <integer> <= 24 and
<empty> ::= <integer> = 24
For <integer> = 0 the program converts it to 24.
```

The binary word is now printed in blocks of <integer> separated by a <space>. The dividing is made from the left.

instruction

The bitpatterns are written as machine instructions including the mnemonic functioncodes.
In case of hardware or software error interrupt No. 0 may occur. This involves an error message:

interrupt no. 0

from the interrupt-sequence. In this case not only the testprograms but even the loader should be stored again.

By activating the RESET and then the START push-button, a jump to the loader is executed, and a new operator-key and -typewriter may be selected.

On the next page is shown some messages to and from the operator during a test.

The testprograms for high-speed devices are so designed that they propose the start address of the input-output buffer by writing:

fbw = <address of first free word>

and waits for input. If the operator types <NL>, he accepts the start address; if he types a slash, the programs ask:

fbw =

and he must input another start address. This address must be within a free part of the core store, i.e. 1) an address lower than the loader 2 start address (but not less than 24) or 2) an address higher than the test-program's top address. If this condition is not fulfiled the program asks for a new input. Contition 1) is only significant when using the relocatable loader with start address greater than 0. After input from the console the address of the last buffer word is calculated and written. If the last buffer word (lbw) is calculated to be without the free part of core store some error messages will occur.

Messages to and from the operator:

```
loader  2
input from dev. no.: 0
rc teletypewriter
device no.  = 16,17,18
channel no. = 22,23
output dev. no. = 5
testprogram:

a   1.2 read (echo)
b   1.3 write key - board
c   1.4 timer
d   2.1 sequence

testprogram: b
number of runs = 1

run no.       1
terminal disconnected


select testprogram: c
number of runs = 10
time,expected:3000-6000 msec
time,measured:
run no.       1


select loader  2
input from dev. no.: 0
rc 315 typewriter
device no.  = 2
channel no. = 7
output dev. no. = 2
testprogram: d
number of runs = 1

sequence =
abcdefghijklmn

run no.       1
abcdefghijklmn
test end

testprogram:

select device no.  = 10
channel no. = 8
output dev. no. = 5
testprogram: b
number of runs = 2

run no.       1
run no.       2
test end

testprogram:
```

THE RELOCATABLE LOADER 2

The relocatable loader consists of the above-mentioned loader and procedures, however so designed that the relocatable loader and the testprograms may be stored everywhere within the available core store if the operator before activating the AUTOLAD push-button puts the start address into w3. This start address must be chosen that

$$0 \leq w3 \leq \text{length of core store} - \begin{array}{l}(\text{length of relocatable loader} + \\ \ \text{length of testprograms})\end{array}$$

All lengths are measured in No. of bytes. The length of the relocatable loader is 2980 bytes.

When w3 = 0, the loader and the testprograms are stored as usually (see chapter 1.1 page 5).