
RCSL No: 52-AA1089
Edition: February, 1982
Author: Allan Bjørn Kristensen

Title:

Upgrade from RC3503 to RC3503C,
Software information

Keywords:

RC3503,
RC3503 Reference Manual 42-i1540,
RC3503 General Information 52-AA1013.

Abstract:

This paper describes the changes made on RC3503 compared to RC3503.

(12 printed pages).

Copyright © 1981, A/S Regnecentralen af 1979
RC Computer A/S

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.

<u>CONTENTS</u>	<u>PAGE</u>
1. SOFTWARE INFORMATION, UPGRADE FROM RC3503 TO RC3503C ..	1
1.1 Module Addressing	1
1.2 Instruction MOB	1
1.3 Instructions ILD, IST	3
1.4 Instruction List	4
1.4.1 INTERLEVEL LOAD	4
1.4.2 INTERLEVEL STORE	5

1. SOFTWARE INFORMATION, UPGRADE FROM RC3503 TO RC3503C

1.

1.1 Module Addressing

1.1

The debugger is changed so addressing of modules mounted in RC3503 are accessed by the addresses:

RAM MODULE decimal No.	MODULE ADDRESS used from debugger	PROM MODULE decimal No.	MODULE ADDRESS used from debugger
0	00C0:	16	00E0
1	00C2:	17	00E2
2	00C4	18	00E4
⋮	⋮	⋮	⋮
15	00DE	31	00FE

Both the new instruction MOB and the old MOV use the normal decimal representation to the module address.

1.2 Instruction MOB

1.2

Format 1a:

F	W	X	D
0	5	7	9 13

Format 1b:

1	1	1	1	0	W	X	1	N	F
0					5	7	9	10	11 15

Format 1c: is illegal.

		F0	F1	F2	F3	F4
MOB	Move block	1	1	1	0	1

Move a block of bytes, or a block of words, from one specified module to another specified module. If $n = 0$, then move bytes, else move words.

The module addresses and the displacement addresses are contained in a parameter block. The first parameter is addressed by "ea". The remaining bytecount is specified by the contents of W1. Note that in word move, count must still be specified as number of bytes.

The parameter block has the form:

```

first parameter addressed by ea    = modules
                                   ea + 2 = destination offset
                                   ea + 4 = source offset

```

The word "modules" contains source module addr. * 256 + destination module addr.

```

Module addresses are:    0-15    for RAM modules
                        16-31    - EPROM modules
                        32-47    - External memory

```

It is only allowed to use the instruction in format a or b. The instruction can be interrupted by debugger request and by external interrupt on higher levels. After interrupt the instruction counter is pointing at the instruction, and W1 is set to the new remaining count. At least one byte is moved before interrupt.

The instruction can be written as:

```

rem_count := W1
if rem_count = 0 then goto fetch
source_module := ((st (ea)/128) and 7E) exor 0C0

```

```

destination_module := (st (ea) * 2) and 7E) exor 0C0

```

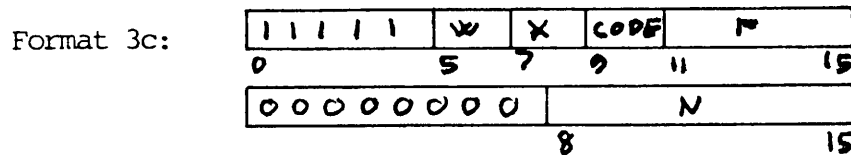
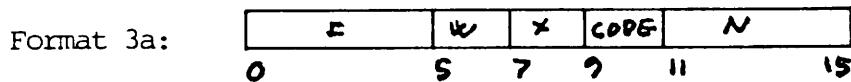
```

destination_addr:= st (ea + 2)
source_addr:= st (ea + 4)
WHILE rem_count > 0 do
  if n = 1 then begin word (dest_module, dest_addr + rem_count):=
    word (source_module, source_addr + rem_count);
    rem_count:= remcount - 1; end;
  ① if higher_level_interrupt then begin
    W1:= rem_count
    W0:= W0-2
    goto fetch
  end;
end; /* WHILE */
goto fetch

```

1.3 Instructions ILD, IST

1.3



		F0	F1	F2	F3	F4
ILD	Interlevel load	1	1	0	1	0
IST	Interlevel store	1	1	0	1	1

The purpose of interlevel instructions is:

1. To transfer information between two working registers belonging to the same or to different interruption levels.
2. To transfer information between a working register to/from the control microprocessor RAM.

① if n = 0 then begin byte (dest_module, dest_addr + rem_count):= byte (source_module, source_addr + rem_count); rem_count:= rem_count - 1; end;

W selects a working register in the current interruption level and X a register belonging to the interruption level specified by the N field, if $N < 124$.

If $N > 127$, then the transfer is between the W reg. pointed out by the W field and the control microprocessor RAM addr. N and 63.

The interlevel instructions may, in addition to their normal execution, also control the interruption system. A letter D, E, or S is then used as a suffix to the basic mnemonic code. The effect of the suffixes and the corresponding CODE field is as follows:

Suffix	Code	Effect on Interruption System
no suffix	00	no change
E	01	enable for all interrupts
D	10	disable for all interrupts
S	11	set the level specified by the N field

Suffix S must not be used if $N > 124$.

1.4 Instruction List

1.4

1.4.1 INTERLEVEL LOAD

1.4.1

- If level < 127 , then
 load the W register, pointed out by the W field, by the contents of the W register, pointed out by the X field, from the level specified by N.
- If level ≥ 127 , then
 load the W register, pointed out by the W field, by the contents of the microprocessor RAM and 255, addr N and 63.

In short form:

```

If  N < 127 then
    W (current level) := X (Nth level);
else W (current level) := Micro RAM (N and 63)

If  code = D then  disable intr. system
If  code = E then  enable intr. system
If  code = S then  set intr. on level N

```

It is not allowed to use S if $N > 124$.

1.4.2 INTERLEVEL STORE

1.4.2

```

If  level < 127, then
    store the W register, pointed out by the W field, in the
    W register, pointed out by the X field, the level speci-
    fied by N.

If  level >= 127, then
    store the W register b8-b15, pointed out by the W field,
    in the microprocessor RAM address N and 63.

```

A store in the microprocessor RAM will be followed by an interrupt on the level pointed out by the micro processor RAM addr. 01, when the micro procesor is ready for a new store. If, however, the instruction is used before the microprocessor is ready, then the insturction will make busy waiting, by decrementing the instruction counter to point on the instruction again. A special case is writing in RAM addr 8. This will force the micro processor to send the contents to the console as an asii character.

Interrupt after such a write will be sent after a pregoing character has been sent.

The instruction in short form:

```
If  N < 127, then
    X (Nth level) := W (current level)
else Micro RAM (N and 63) := W (current level)
```

```
If  code = D then  disable intr. system
If  code = E then  enable intr. system
If  code = S then  set intr. level
```

It is not allowed to use S if $N > 124$.