NORD-50 MONITOR

User's Guide

and

System Documentation

NORSK DATA A.S

NORD-50 MONITOR

User's Guide

and

System Documentation

REVISION RECORD

Revision	Notes
11/76	Original Printing
11/78	Second Edition – replaces original printing

NORD-50 MONITOR -- User's Guide and System Documentation
Publication No. ND-60.076.02

NORSK DATA A.S.
Lørenveien 57, Postboks 163 Økern, Oslo 5, Norway

PREFACE

This manual is recommended as a necessary documentation to any programmer/ system analyst intending to obtain information about the implementation and operation of the NORD-50 Monitor.

The first five chapters and Appendix A and B are intended to give a description of how to run NORD-50 programs and how to implement the NORD-50 Monitor to run under control of the SINTRAN III operating system in a NORD-10/NORD-50 computer system installation.

It is recommended that the reader of this manual should have available the "NORD-50 FORTRAN Reference Manual", and should have attended the course US03 — Introduction to SINTRAN III or at least have obtained the same knowledge about operating as Time-sharing user under SINTRAN III.

Chapter 6 is intended to give more detailed information about how the NORD-50 Monitor is built up and how it operates. This information is addresses to supervisory and staff personnel as well as system programmers and analysts.

TABLE OF CONTENTS

+ + +

Section: *Page:*

1 INTRODUCTION

1.1 *GENERAL*

The NORD-50 Montitor is a system for running programs on NORD-50 under SINTRAN III in interactive or batch mode of operation. The system is entered by the command NORD-50, given to the SINTRAN III background command processor (@). If NORD-50 is in use, the message ALREADY IN USE is printed, and the monitor is not entered.

When the system is ready, it prints an *. A set of commands is then accepted. The commands may be abbreviated as in SINTRAN III. Missing arguments are requested.

1.2 *ENVIRONMENT*

For a better understanding of the architecture of the NORD-50 Monitor, it may be interesting to review the design ideas and implementation involved.

The NORD-50 is a high performance digital computer to be used as an auxiliary computer, or computer module, in a general purpose computing system.

The NORD-50 is always controlled from a NORD-10 computer system, where NORD-10 runs the operating system and prepares the jobs for execution in the NORD-50.

NORD-10 and NORD-50 are connected to a shared memory, each with its own channel, by use of a multiport memory system.

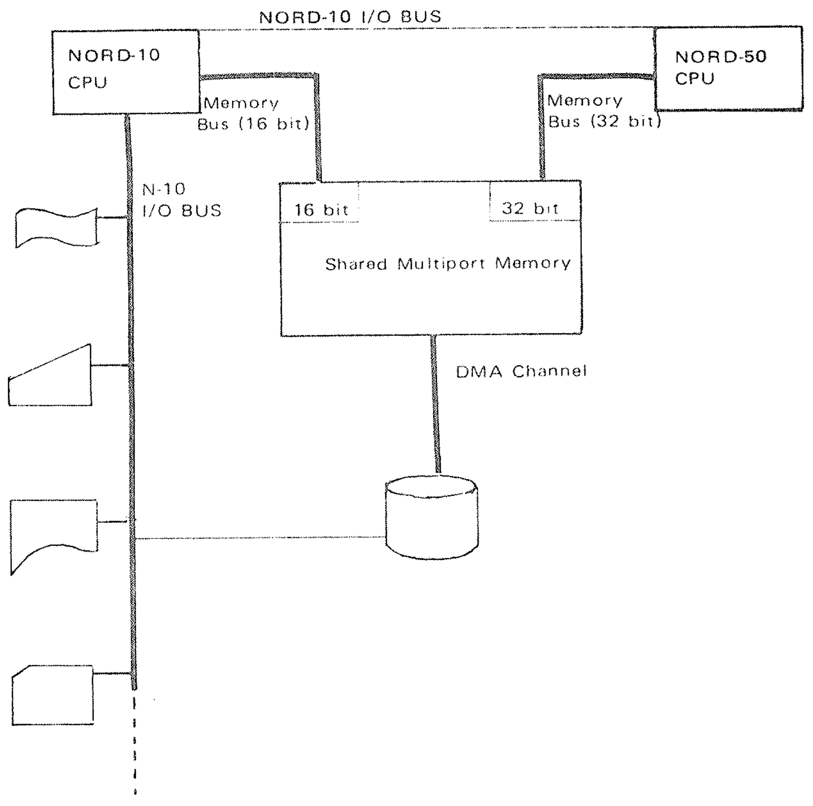


Figure 1.1: Typical Configuration of a NORD-10/NORD-50 Computer System

The communication between the NORD-10 and the NORD-50 (for controlling the NORD-50) is based on the use of the IOX instruction in NORD-10. Accordingly, the hardware part of the communication unit in NORD-50 is made in such a way that the NORD-10, when software is concerned, may regard the NORD-50 as an I/O device. In the communication procedure, the NORD-10 has complete control and the NORD-50 is regarded as a slave to the NORD-10. All the data (register contents) transfers between the two computers are done in 16 bit parallel mode. This means that the NORD-10 must use two IOX instructions to transfer a NORD-50 word to or from the NORD-10.

Note: 1 page is 1024 16 bit words or 512 32 bit words. 1K is 1024 words (16 or 32 bits specified).

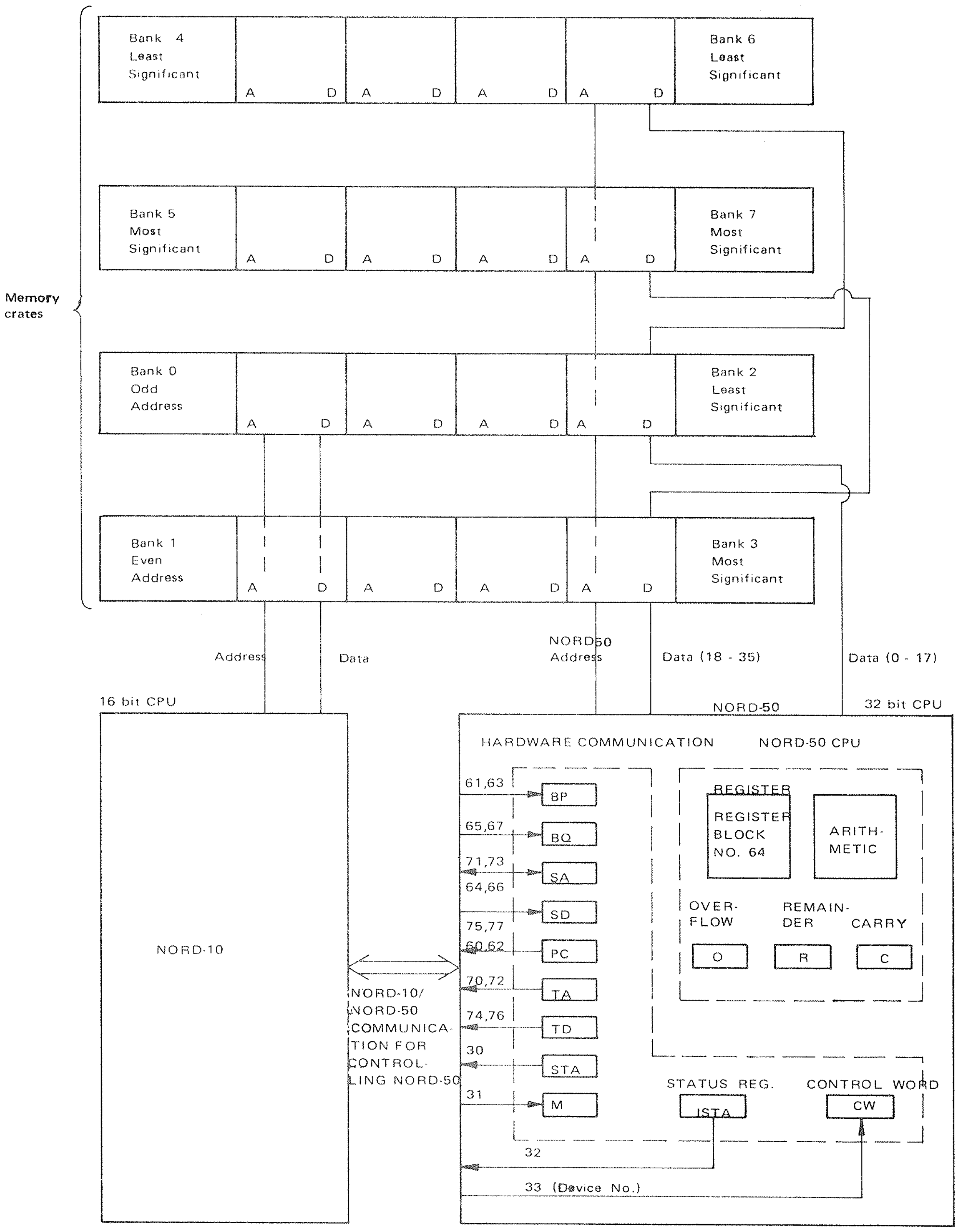


Figure 1.2: Typical Hardware Connection of a NORD-10/NORD-50 System

Definition of the registers in the preceding figure:

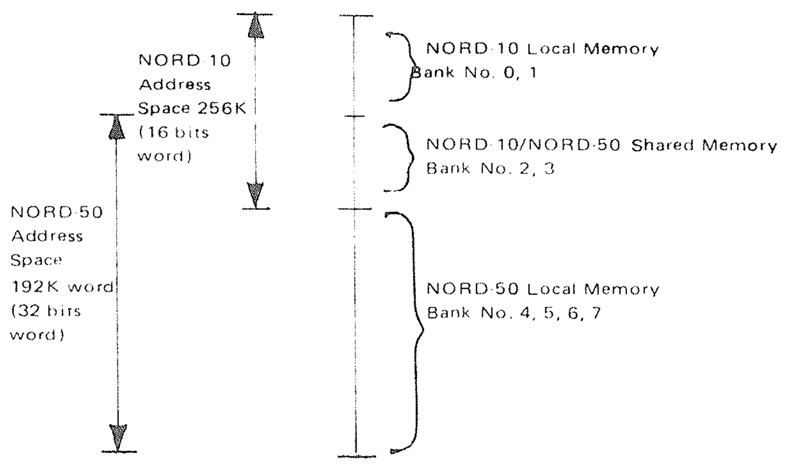
BP	Breakpoint Register No. 1 (32 bits)	Registers read/written to/from NORD-10 by IOX inststructions in NORD-10
BQ	Breakpoint Register No. 2 (32 bits)	
SA	Simulated Address (32 bits)(start address)	
SD	Simulated Data (32 bits)	
PC	NORD-50 Program Counter	
STA	NORD-50 Status Register	
TA	Test Address Register (last memory reference)	
TD	Test Data Register (last data to/from memory)	
M	Modus Register (to control the operation of N-50)	
ISTA	Interface Status Register	Standard NORD-10 I/O Interface reg. for controlling an I/O device (N-50)
CW	Interface Control Word Register	
O	Overflow Register in NORD-50	
R	Remainder Register in NORD-50	Reg. read/written to/from memory by the NORD-50 save/unsave routines (placed in NORD-50 memory)
C	Carry Flip-Flop in NORD-50	
REG	Register Block in NORD-50	

In Figure 1.2, the device numbers used by NORD-10 to access the different registers in the NORD-50 communication modules are shown.

Before NORD-50 is started to run a program, NORD-10 has to set the actual registers in the communication module, for instance, the program start address in memory, the operation modules to be used by NORD-50 (SA and M registers, etc.).

Then NORD-50 is started by setting the activate bit in the Interface Control Word Register (CW) and if one wishes an interrupt to be generated when NORD-50 stops, the interrupt enable bit must also be set. The NORD-50 communication module is connected to interrupt level 12 in NORD-10.

In the examples, Figure 1.2, the memory is used in the following way:


NORD 10 Address Space 256K (16 bits word)
NORD 50 Address Space 192K word (32 bits word)
NORD 10 Local Memory Bank No. 0, 1
NORD 10/NORD 50 Shared Memory Bank No. 2, 3
NORD 50 Local Memory Bank No. 4, 5, 6, 7

For further information, refer to the manuals "NORD-50 Reference Manual" and NORD-10/NORD-50 Communication System".

2 AN EXAMPLE OF HOW TO RUN A PROGRAM ON NORD-50

Programs for NORD-50 must be written in NORD-50 assembler or NORD-50 FORTRAN language.

The NORD-50 assembler for SINTRAN III, the NORD-50 FORTRAN Compiler and the NORD-50 Loader are described in the manuals – "Assembler for NORD-50", "NORD-50 FORTRAN Reference Manual" and "NORD-50 Loader User's Guide".

How to run a program on NORD-50:

Example:

```
@N50FTN
– NORD-50 FTN COMPILER –
$PROGRAM-MAP 14646
$CROSS-REFERENCE


$COM SOURCE, 1, OBJ: BRF5
    1*        PROGRAM NORD-50
    2*        WRITE (1, 2)
    3*   2    FORMAT (*THIS PROGRAM EXECUTES IN NORD-50*)
    4*        END

– – – – – – – – – – – MEMORY ADDRESS MAP – – – – – – – – – – –
LINE:      +0        +1      +2      +3      +4  +5   +6   +7   +8   +9
    0*              14707   14715           14736

– – – – – – – – – – – – – LOCAL IDENTIFIERS – – – – – – – – – – – – –
    NONE
              TOTAL LOCAL DATA SPACE 211 (OCTAL)

– – – – – – – – – – – – –COMMON IDENTIFIERS – – – – – – – – – – – –
    NONE

– – – – – – – – – – – –EXTERNAL REFERENCES – – – – – – – – – – – –
    INTEGER FIO.

– – – – – – – – – – – –CROSS-REFERENCE MAP – – – – – – – – – – – –

NORD50     1

    5*        EOF

5 STATEMENTS COMPILED
CPU-TIME USED IS 0.9 SEC.
$EXIT
@N50 LDR
NORD-50 LOADER – F
MEMORY-IMAGE FILE:   IMAGE:NOR5
*LOAD OBJ
FREE:     0014707     077777
*EXIT
FREE:     0014707     077777
@NORD-50
```

```
NORD-50 MONITOR
*PLACE IMAGE ↓
*RUN ↓
THIS PROGRAM EXECUTES IN NORD-50
—** END *** — AT: 000015 —
*EX ↓
@
```

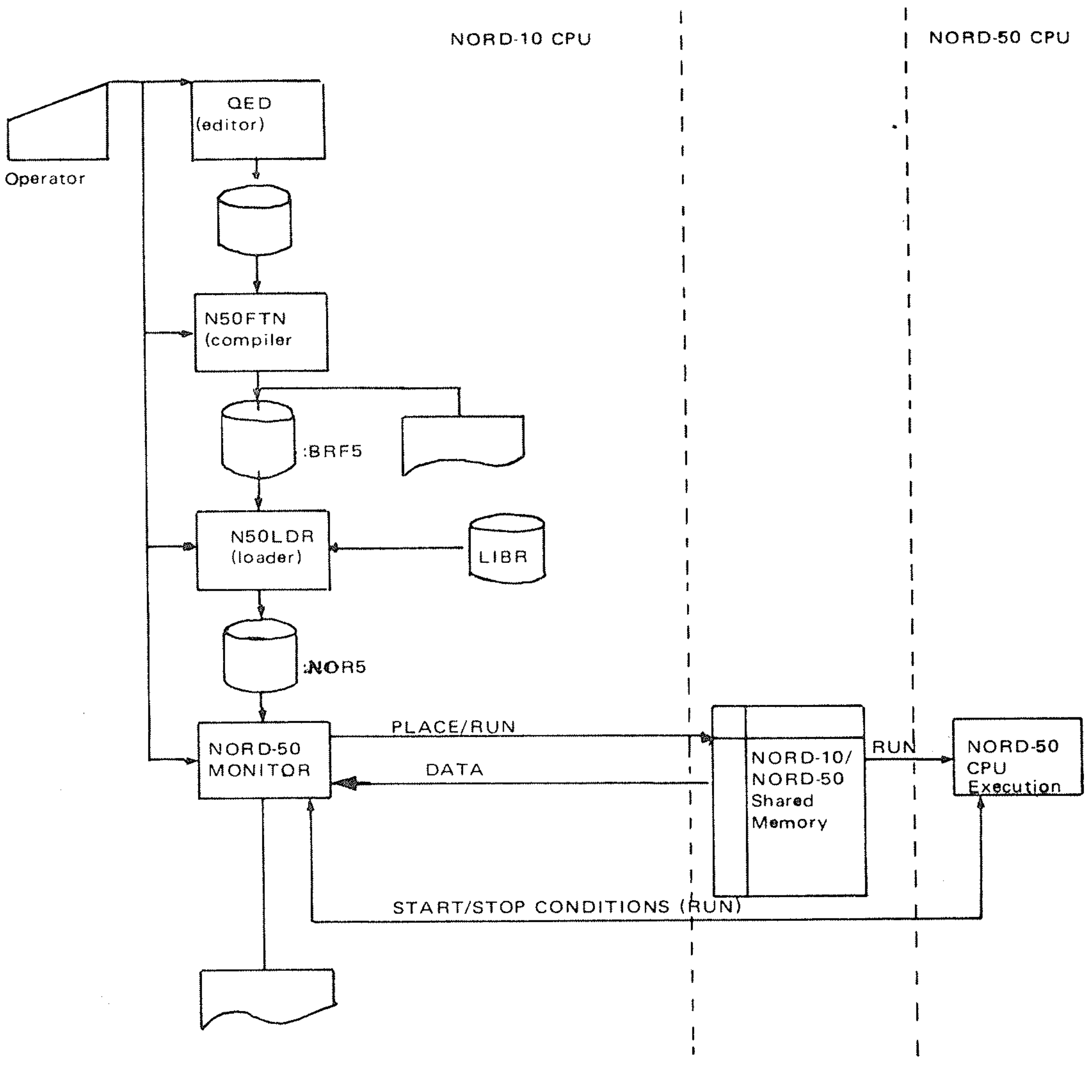


Figure 2.1: Example (Flowchart) of Preparing and Running NORD-50 Program

Appendix A lists all NORD-50 Monitor commands available for the operator, while Appendix B lists all the SINTRAN III Monitor calls and FORTRAN routines that may be called from a NORD-50 program.

3 THE PROGRAMMER'S INTERACTION WITH THE SYSTEM

3.1 *OPERATOR COMMANDS*

3.1.1 *Command Syntax and Parameters*

The NORD-50 Monitor is entered by the command @NORD-50, given to the SINTRAN III background command processor. If NORD-50 is in use, the message ALREADY IN USE is printed and the monitor is not entered.

A NORD-50 CPU time limit in minutes may be given as a parameter to the NORD-50 command.

In batch mode this time limit will override the time limit given in the ENTER command. For systems with multiple NORD-50 CPUs the CPU number may be given as a paramter to the NORD-50 command

@NORD-50 [<CPU no.>] [,<CPU time limit>]

If the CPU number is not specified in systems with multiple NORD-50 CPUs the first free CPU will be used.

When the system is ready, it prints * and a set of commands is then accepted. The commands may be abbreviated as in SINTRAN III. Missing arguments are asked for. Arguments to the commands may be:

<access mode>	file access mode code as in SINTRAN III File System or "READ" or "WRITE"
<address>	NORD-50 memory addresses. Octal default.
<break conditions>	A, F, S, D, O, U, P or combinations
<connected file number>	decimal default
<file name>	
<file number>	octal default
<formats>	the characters O, I, D, A, F, S, B, T, C or combinations
<register>	NORD-50 register octal default number preceded by the character R
<op. code>	NORD-50 assembler op. code or pseudo op. code ACN or FCN
<program unit>	FORTRAN program unit name. The program has to be compiled in debug mode (command DEBUG in the NORD-50 FORTRAN Compiler). The specified program unit is default until another unit is specified.

<line number>	FORTRAN source program line number. The line number printed on the source program listing from the compiler or the line number in QED. The program has to be compiled in debug mode.
<variable name>	FORTRAN variable name, array names or array element. The array indexes have to be integer constants if any array element is given. The program has to be compiled in debug mode.

Default mode for numbers may be overridden by ending the number by B for octal or D for decimal numbers.

If the first character in a command line is @, the line is taken as a SINTRAN III command.

Control is returned to the SINTRAN III command processor when the character "control L" or the command EXIT is typed.

Typing "escape" when in the NORD-50 Monitor may given two different actions:

1. NORD-50 is active executing a program. The program is stopped. Execution will continue when the RUN command is given. *Open files are not closed automatically.*

2. NORD-50 Monitor is in interactive command mode. The command being typed is ignored and the NORD-50 Monitor command processor is restarted.

The operator commands are listed in Appendix B.

3.1.2 *Debug*

3.1.2.1 FORTRAN Debug

When a program is compiled in debug mode (command DEBUG in the NORD-50 FORTRAN compiler) information about source program line numbers and variable names are included in the binary relocatable output from the compiler.

The NORD-50 LOADER assembles this information into a debug information table on the executable format file produced by the loader.

The NORD-50 MONITOR uses the debug information table to get information about the addresses of source program line numbers or variables, the types of the variables and the dimension of arrays.

Breakpoints and exhibit points are implemented by a monitor call (51_8) while the trap feature uses the NORD-50 breakpoint registers and break conditions.

For program units with debug information available to the NORD-50 Monitor the commands with the parameter types <program unit>, <line no.> and <variable name> can be used. All other (debug) commands can be used for any program.

3.1.2.2 Breakpoints

There are 3 types of breakpoints in the NORD-50 Monitor.

1. Breakpoints
2. Conditional breakpoints
3. Exhibit points

When the execution of the program reaches a breakpoint (1) the execution stops, and the control is given to the command input. When a RUN command is issued the execution continues until the same or another breakpoint is reached. When the execution of the program reaches a conditional breakpoint (2), a condition on a variable is checked. If the condition is true, the execution will stop as in a breakpoint, otherwise the execution will continue as if no breakpoint were present. Exhibit points (3) never cause a break, but when the execution of the program reaches an exhibit point the value of a variable is displayed on the terminal or on an exhibit output file (or device).

The NORD-50 Monitor can handle up to 8 breakpoints, conditional breakpoints and/or exhibit points. The command BREAK-NUMBER sets the current breakpoint number and the following break/exhibit point commands will change, move or set this breakpoint. When the program execution stops in a breakpoint the current breakpoint number is set to the breakpoint. The CLEAR-BREAK command will set the current breakpoint number to one and the RESET-BREAK command will set the current breakpoint number to the one restored by the command.

3.1.3 *Commands for Taking a Program in Executable Format on a File and Placing it in NORD-50 Memory and Starting it*

3.1.3.1 PLACE <file name or file number>

Moves the executable NORD-50 program file prepared by the NORD-50 loader into the NORD-50 memory. Default file type: NOR5.

NORD-50 memory is allocated according to the break conditions set from the NORD-50 Loader when the core image file is prepared. See NORD-50 Loader and the command BREAK-CONDITIONS.

3.1.3.2 LOAD–AND–GO <file name or file number>

Equals the commands PLACE and RUN performed in one operation.

3.1.3.3 RUN

Starts execution of the NORD-50 program in the main program start address. Continues execution after a break. Continues execution after a program is stopped by "escape".

3.1.3.4 GOTO <address>

Starts execution of NORD-50 program in the specified address.

3.1.4 *Commands for Opening and Connecting Files*

3.1.4.1 OPEN-FILE <file name>, <connected file number>, <access mode>

Opens a file and connects a file number used in the program.

Access modes:

R	sequential read
W	sequential write
X	random access
A	append
C	common access
D	direct file transfer

Combinations:

Sequential Read/Write:

RW	sequential read/write (READ/WRITE, INBT/OUTBT)
WA	sequential write append

Random Read/Write:

RX	random read (RFILE)
WX	random write (WFILE)
RC	random read with read and write access allowed fromo the users
WC	random read/write with read/write access allowed from other users

Direct Read/Write:

D	direct transfer modus 8 when opened from NORD-50 programs (see Sectioon 3.3)
DC	direct transfer with the file closed, modus 9 from NORD-50 programs (see Section 3.3)

System Selected Access:

READ The system will select the access mode R, RX or D. The most optimal access mode which can be used for FORTRAN READ statement for the file/device is selected. For example:

terminal; R, tape reader; R, indexed file; RX, continuous file; D, magnetic tape; D.

WRITE The system will select the access mode RW, WX or D. Refer to READ access above.

3.1.4.2 CLOSE-FILE <connected file number>

Closes a file and disconnects the file number.

3.1.5 *Commands for Debugging*

A program used for some of the examples below:

```
@N50-FORTRAN ↓
— NORD-50 FTN COMPILER 77.11.24 —
$DEBUG ↓
$COM SAMPLE, 1, OBJ ↓
 1*        PROGRAM SHOW
 2*        INTEGER I, J, K
 3*        REAL X, Y
 4*        I = 1
 5*        J = −2
 6*        K = I − J
 7*        X = 1.24
 8*        Y = 2.56
 9*        CALL SUBR (I, X)
10*        Y = Y + X
11*        END
12*
13*        SUBROUTINE SUBR (I, X)
14*        I = I*X + X
15*        CALL DEBUG (I)
16*        END
17*
18*        SUBROUTINE DEBUG (I)
19*        IF (I.GE.128) THEN
20*            I = 3
21*        ENDIF
22*        END
23*        EOF

20 STATEMENTS COMPILED, OCTAL SIZE = 144
CPU-TIME USED IS 1.4 SEC.
```

3.1.5.1 BREAK-LINE <line no.> [<program unit name>]

Sets a breakpoint in the line number specified or moves the breakpoint to the line number.

Default program unit name is the last unit name given in any previously issued command.

Example:

```
*BREAK-LINE 7,SHOW ↓
*RUN ↓
— BREAK NO.    1 SHOW    LINE NO.    7    — AT: 016255 —
```

3.1.5.2 BREAK-ADDRESS <address>

Sets a breakpoint in the NORD-50 address specified or moves the breakpoint to the address.

3.1.5.3 C

Single instruction execution. The command may be used when a breakpoint is reached in the program. The breakpoint is moved one step and the program is restarted. A warning is given if the next instruction may cause a jump or a skip.

3.1.5.4 CONDITIONAL-BREAK-LINE <line no.> <program unit name> <variable name> <program unit name> <low limit> (<high limit>)

Sets (moves) a conditional breakpoint in (to) the line number specified. When a conditional breakpoint is reached during program execution the specified variable is checked against the limits. If low limit ⩽ variable ⩽ high limit, the execution is broken and commands can be issued, otherwise the program execution continues as if no breakpoint were present.

Input formats for the limits are type dependent and like a standard FORTRAN format.

For type logical and complex only the low limit should be given and the test is: break if "low limit" = variable. There is no conditional break for type character.

Example:

```
*CONDITIONAL-BREAK-LINE 10,,I,,20,40 ↵
BREAKPOINT NO.      1 CHANGED
```

3.1.5.5 CONDITIONAL-BREAK-ADDRESS <address of break> <address of variable> <low limit> <high limit>

This command is similar to the command CONDITIONAL-BREAK-LINE, but the breakpoint and the variable is given as octal addresses and the variable is assumed to be of type integer (refer to Section 3.1.5.4).

The command is intended for use with assembly programs or with FORTRAN program units compiled with the debug option off.

3.1.5.6 EXHIBIT-LINE <line no.> <program unit> <variable> [<program unit>]

Sets (moves) an exhibit point in (to) the line number specified. When an exhibit point is reached during program execution the variables value is written to the terminal or to the file specified by an EXHIBIT-OPEN command (see Section 3.1.5.8).

The output format used depends on the variable type. Additional formats can be requested by the X-FORMAT command (see also Section 3.1.5.30).

Example:

```
*EXHIBIT-LINE ↵
LINE NO.: 6 ↵
PROGRAM UNIT NAME: ↵
VARIABLE NAME: J ↵
BREAKPOINT NO.        1 MOVED
```

3.1.5.7 EXHIBIT-ADDRESS <address in program> <address of variable>

Sets (moves) an exhibit point in (to) the address specified. When an exhibit point is reached during program execution the variables value is written to the terminal or to the file specified by an EXHIBIT-OPEN command (see Section 3.1.5.8).

The output format used depends on the variable type. Additional formats may be requested by the X-FORMAT command (see Section 3.1.5.30).

3.1.5.8 EXHIBIT-OPEN <output file name>

Opens a file for output from exhibit points. If no EXHIBIT-OPEN command is given, the exhibit output goes to the terminal.

3.1.5.9 EXHIBIT-CLOSE

Closes the exhibit output file. The exhibit output file is also closed at the end of a program or when a CLOSE-1 command or monitor call is executed.

3.1.5.10 BREAK-NUMBER <breakpoint number>

Sets breakpoint number (1 to 8) for the breakpoint, conditional breakpoint and exhibit point commands. The breakpoint number is also changed when the program execution stops in a breakpoint.

All 8 break/exhibit points can be active at the same time.

3.1.5.11 RESET-BREAK <breakpoint number>

Restores a breakpoint or an exhibit point.

3.1.5.12 CLEAR-BREAKS

Restores all breakpoints, exhibit points and the trap.

3.1.5.13 LIST-BREAKS

Lists some information about the active breakpoints, conditional breakpoints, exhibit points and trap on the terminal.

Example:

```
*LIST-BREAKS ↓
NO.  1     0016247     SHOW        LINE NO.   4
 – EXHIBIT      0016310:      J
NO.  2     0016253     SHOW        LINE NO.   6
NO.  3     0016251     SHOW        LINE NO.   5
 – CONDITION    0016310:   J     LOW LIMIT    10         HIGH LIMIT    40
```

3.1.5.14 TRAP-LINE <low line no.> <high line no.> <program unit> <conditions>

Sets a trap in executable statements. If the access conditions are violated the program execution will stop and the message ADDRESS VIOLATION will be displayed (see also Section 3.1.5.33).

3.1.5.15 TRAP-VARIABLE <variable name> <program unit> <conditions>

Sets a trap on access to the variable. If the access conditions are violated the program execution will stop and the message ADDRESS VIOLATION will be displayed.

The parameter <variable name> may be an array name or an array element. In the case of an array name all accesses to the array are checked against the access conditions while in the case of an array element only the accesses to the element specified is checked. (Refer also to Section 3.1.5.33.)

Example:

```
*TRAP-VARIABLE K ↓
PROGRAM UNIT NAME: ↓
BREAK CONDITIONS (A, F, D, S, O, U, P, N): S ↓
*RUN ↓
– ADDRESS VIOLATION – AT: 016254 –
```

3.1.5.16 TRAP-ADDRESS <low address> <high address> <conditions>

Sets a trap on access to a memory area. If the access conditions are violated the program execution will stop and the message ADDRESS VIOLATION will be displayed (refer to Section 3.1.5.33).

3.1.5.17 RESET-TRAP

Restores the trap set.

3.1.5.18 NEST

This command lists the current dynamic subroutine call nesting on the terminal. Program unit names and octal addresses of the call statement are always listed. If debug information is also available for the unit the FORTRAN source program line numbers of the call statements are listed.

Example:

```
*BREAK-LINE 19,DEBUG ↵
*RUN ↵
– BREAK NO.      1    DEBUG      LINE NO.      19      – AT: 016363
*NEST ↵
UNIT          ADDRESS        LINE NO.
DEBUG         0016363             19
SUBR          0016334             15
SHOW          0016261              9

*
```

3.1.5.19 LIST-UNITS [<output file>]

LIST-UNITS lists the names, first addresses in memory and first line numbers on the source file for the program units with debug information available.

Example:

```
*LIST-UNITS ↵
NAME          ADDRESS        LINE NO.
SHOW          00000016241          1
SUBR          00000016317         13
DEBUG         00000016361         18
```

3.1.5.20 ENTRIES-DEFINED [<output file>]

ENTRIES-DEFINED lists the subroutine entry points and common labels with octal addresses.

3.1.5.21 ENTRIES-UNDEFINED [<output file>]

ENTRIES-UNDEFINED lists the undefined entries, if any.

3.1.5.22 DISPLAY <variable name> [<program unit>]

DISPLAY lists the address, type and value of the variable on the terminal. The output format for the value is dependent on the type.

Example:

```
*DISPLAY I ↙
016407:     INTEGER       VARIABLE I          0
```

3.1.5.23 CHANGE <variable name> [<program unit>] <new value>

This command changes the value of the variable. The input format for the new value is type dependent free format.

Example:

```
*B–L     10,SHOW ↙
*RUN ↙
– BREAK NO.      1     SHOW       LINE NO.    10        – AT: 016266
–
*DIS   I ↙
016307:     INTEGER       VARIABLE I              2
*CHANGE I,,30 ↙
*DISPLAY   I ↙
016307:     INTEGER       VARIABLE I             30
```

3.1.5.24 LOOK-AT <address or register>

or
<address>
or
<register>

LOOK-AT enters the "look-at mode" to display and/or changes the NORD-50 memory or registers. Output formats are set in the command FORMAT. In the "look-at mode" the system will respond like this:

<op. code>, <number>, -----, <number>

or

<number>↵	contents of memory or register is changed
<address or register> /	contents are displayed
↵	the next memory location or register is displayed
@	returns to normal command mode

Refer to the example in Section 3.1.3.27.

3.1.5.25 DUMP-ARRAY <array name> <program unit> [<output file>]

This command dumps the values of all the array elements on the output file. The output format is dependent on the array type. An array element can be specified. The part of the array from this element and up will then be printed. The output can be broken by pushing the escape key.

3.1.5.26 DUMP-VARIABLES <program unit name> [<output file>]

This command dumps the name, type, address and value of all the variables in the program unit. The format is as for the command DISPLAY.

Example:

```
*DUMP-VARIABLES ↵
PROGRAM UNIT NAME: ↵
016307:        INTEGER       VARIABLE I                    30
016310:        INTEGER       VARIABLE J                    –2
016311:        INTEGER       VARIABLE K                     3
016313:        REAL          VARIABLE X          1.24000E + 00
016315:        REAL          VARIABLE Y          2.56000E + 00
```

3.1.5.27 DUMP-ALL-VARIABLES [<output file>]

This command dumps the name, type, address and value of all the variables in all program units with debug information available on the output file.

Example:

```
*DUMP-ALL-VARIABLES ↲

*** PROGRAM UNIT ***       SHOW

016307:       INTEGER       VARIABLE I                  30
016310:       INTEGER       VARIABLE J                  −2
016311:       INTEGER       VARIABLE K                   3
016313:       REAL          VARIABLE X         1.24000E+00
016315:       REAL          VARIABLE Y         2.56000E+00

*** PROGRAM UNIT ***       SUBR

016307:       INTEGER       VARIABLE I                  30
016313:       REAL          VARIABLE X         1.24000E+00

*** PROGRAM UNIT ***       DEBUG

016407:       INTEGER       VARIABLE I                   0
```

3.1.5.28 DUMP-ADDRESS <low address or register>, <high address or register>, <file name or number>

The contents of the address or register interval is printed on the specified file. Output formats used are determined by the last FORMAT command given.

Example:

```
*FORMAT OI ↓
*DUMP-ADDRESS ↓
LOW ADDRESS: 0 ↓
HIGH ADDRESS: 10 ↓

000000:  20227050000          STR    05,   0000,   04,   00,   I
000001:  00223050000          LDR    05,   0000,   04,   00
000002:  00267040001          STR    04,   0001,   05,   00
000003:  00267060002          STR    06,   0002,   05,   00
000004:  00201000001          RTJ    00,   0001,   04,   00
000005:  00012060013          JRZ    06,   0013,   00,   00
000006:  14000070006          RAD    07,   00,     06
000007:  00263050000          LDR    05,   0000,   05,   00
000010:  00263040001          LDR    04,   0001,   05,   00
*FORMAT ↓
FORMATS (O, I, D, A, F, S):   DOSA ↓
*D-AD 0 10 1 ↓
000000:  20227050000            -210789424   \P         -3.18891E-75
000001:  00223050000              38555648   LP          2.63619E-75
000002:  00267040001              47988737   \@          1.27470E-74
000003:  00267060002              47996930   \           1.27643E-74
000004:  00201000001              33816577               1.17452E-75
000005:  00012060013               2646027   (           7.04220E-78
000006:  14000070006            1610641414               1.71305E+38
000007:  00263050000              46944256   LP          1.05448E-74
000010:  00263040001              46940161   L@          1.05361E-74

*LOOK-AT 0 ↓
000000:  20227050000           -2107879424   \P         -3.18891E-75 ↓
000001:  00223050000              38555648   LP          2.63619E-75@
*FORMAT OI ↓
*0 ↓
000000:  20227050000          STR    05,   0000,   04,   00,   I   LDR 1 5 ↓
000001:  00223050000          LDR    05,   0000,   04,   00        STR 1 6 ↓
000002:  00267040001          STR    04,   0001,   05,   00        STOP 134 ↓
000003:  00267060002          STR    06,   0002,   05,   00        177 ↓
000004:  00201000001          RTJ    00,   0001,   04,   00          ↓
000005:  00012060013          JRZ    06,   0013,   00,   00        FCN 2.334 ↓
000006:  14000070006          RAD    07,   00,     06              5/
000005:  10042530040          EXC    53,   0040,   01,   10        @
*FO S ↓
*5 ↓
000005:  2.33400E+00          @
*
*EXIT ↓
@
```

3.1.5.29 FORMAT <formats>

Sets output formats to be used in the commands DUMP-ADDRESS and LOOK-AT. Formats may be O, D, F, S, I, A, B, T, C or any combination of these characters.

O	octal
D	decimal
F	floating point (64 bits)
S	floating point (32 bits)
I	instructions (dissassembled)
A	ASCII (one word = 4 ASCII characters)
B	binary
T	two 16-bits octal numbers (NORD-10 compatible output)
C	four 8 bit octal numbers

If the FORMAT command is not used formats are defaulted to OI. Refer to Section 3.1.5.28.

3.1.5.30 X-FORMAT <formats>

Sets output formats to be used by the DISPLAY, DUMP-VARIABLES and DUMP ALL-VARIABLES commands in addition to the type dependent default output format. For the parameter <format> refer to Section 3.1.5.29.

3.1.5.31 STATUS

STATUS prints some information about the NORD-50 status. This command is useful when NORD-50 stops because of any error condition (refer to the example in Section 3.1.6.3).

3.1.5.32 SAVE <file name or file number>

The NORD-50 memory, registers and status are saved on the specified file. The areas 0 < BP and BQ < maximum memory address are saved. The contents of BP and BQ breakpoint may be changed by the command BREAK-CONDITIONS.

Breakpoints, exhibit points and traps are removed before the program is saved.

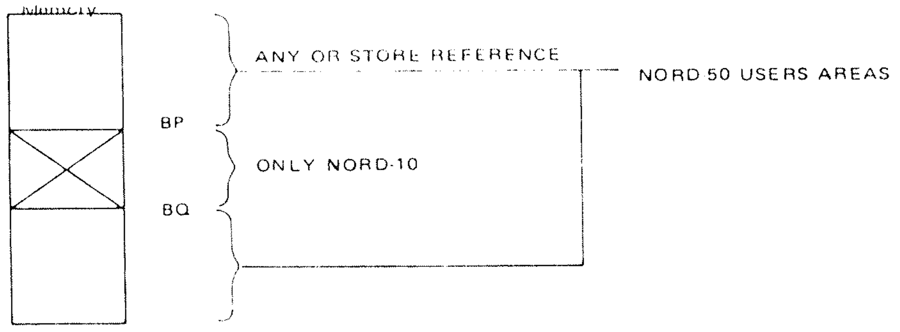
3.1.5.33 BREAK-CONDITIONS <BP address>, <BQ address>, <break conditions>

Set the NORD-50 breakpoint registers and break conditions. <Break conditions may be A, D, F, S, O, U, P or any combination of these characters.

A	stop on any reference in BP ⩽ BQ
D	stop on data reference in BP ⩽ BQ
S	stop on store reference in BP ⩽ BQ
F	stop on fetch reference in BP ⩽ BQ
O	stop on overflow
U	stop on underflow
P	stop on parity error in memory

The BREAK-CONDITIONS command also affects the memory allocation for NORD-50. If BP < BQ and the condition A or S (any or store reference) is on, some of the memory in address interval [BP, BQ> is free, otherwise all the NORD-50 memory is reserved for NORD-50.

Example:



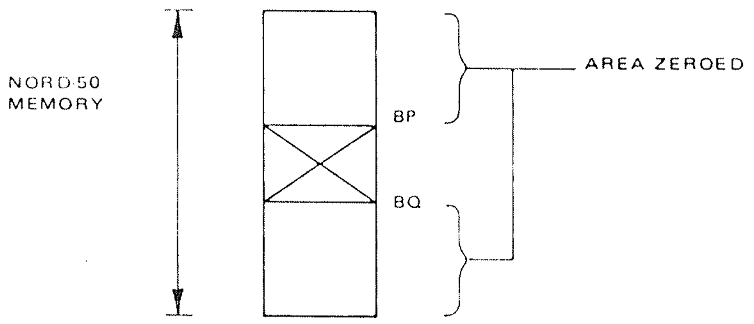
Example:

```
*BREAK-CONDITIONS ↓
BREAKPOINT REGISTER 1 (BP): 20000 ↓
BREAKPOINT REGISTER 2 (BQ): 200000 ↓
BREAK CONDITIONS (A, F, D, S, O, U, P): S ↓
*LIST-MEMORY 1 ↓
SEGMENT NO.     FIRST ADDR.     SIZE     IN USE   TYPE
      061          000000       0020        X     DYNAMIC
      046          020000       0020              DYNAMIC
      047          040000       0020              DYNAMIC
      050          060000       0020              DYNAMIC
      051          100000       0020              DYNAMIC
      052          120000       0020              DYNAMIC
      053          140000       0020              DYNAMIC
      054          160000       0020              DYNAMIC
      055          200000       0020        X     DYNAMIC
      056          220000       0020        X     DYNAMIC
      057          240000       0020        X     DYNAMIC
      060          260000       0020        X     DYNAMIC
     NONE          300000       0020        X     LOCAL MEMORY
                                                  FOR NORD-50

NORD-10 PAGE NO. FOR NORD-50 ADDRESS 0:           0100
```

3.1.5.34 ZERO-MEMORY

This command stores zero in NORD-50 memory in the areas outside the BP, BQ area. If break-conditions O, U or P occur the whole memory is zeroed.



3.1.6 *Commands for Performance Measurements*

3.1.6.1 HISTOGRAM-ON <first word address>, <words/channel>, <no. of channels>

This command clears the Histogram and starts sampling of PC (program counter).

3.1.6.2 HISTOGRAM-OFF

This command stops the program counter sampling.

3.1.6.3 HISTOGRAM-WRITE <output file>

This command prints the histogram.

Example:

```
@NORD-50 ↓

NORD-50 MONITOR
*PL FTEST ↓
*HISTO-ON ↓
FIRST WORD ADDRESS (<200000B): 0 ↓
WORDS/CHANNEL (OCT): 1000 ↓
NUMBER OF CHANNELS (< = 100B): 20 ↓
*OPEN TELE 5 W ↓
*RUN ↓
NEW RUN DATA? (TYPE 0 OR 1) 0 ↓
     181 DATA READY
                        1 RUN(S) COMPLETED
END. . .DO YOU WANT RESTART? (0 OR 1): 0 ↓
— *** END *** — AT: 000015 —
*HIS-WRI ↓
FILE NAME OR NUMBER: 1 ↓
TOTAL COUNTS:          127616
COUNTS IN TABLE:    127616          (100.00 % OF TOTAL)

PERCENTS OF COUNTS IN TABLE:

                              000000      001000      002000      003000

000000                         16.25        0.03        0.12        0.00
004000                          0.00           0           0           0
010000                          5.26        0.86       77.44           0
014000                             0           0           0           0
*STATUS ↓
000015/       000000000134          STOP          000134
PROGRAM STOP
LAST START ADDRESS:                                                000671
LAST MEMORY REFERENCE ADDRESS:                               00000000016
LAST DATA TO/FROM MEMORY:                                    00027070205
BREAKPOINT REGISTER 1 (BP):                                        012547
BREAKPOINT REGISTER 2 (BQ):                                        254347
BREAK CONDITIONS SET:
/ PARITY ERROR / ANY REFERENCE /
```

3.1.6.4 CPU-TIME

Prints the NORD-50 CPU time used since NORD-50 Monitor was entered.

3.1.7 *Commands for Hardware and Software Facilities*

3.1.7.1 HARDWARE-GOTO <address>

Starts execution of a NORD-50 program in the specified address. The NORD-50 register block is not saved when this command is used and therefore the register contents cannot be examined. This command is usually used when debugging the NORD-50 hardware.

3.1.7.2 MASTER-CLEAR

Brings the NORD-50 out of any hang-up state.

3.1.7.3 HARWARE-STATUS

Prints some information about the NORD-50 hardware status: STATUS, PC, SA, TA, TD (actually read from hardware).

3.1.7.4 CARRY

Prints the block 0 variable "carry" in which the contents of NORD-50 "carry flip-flop" is saved.

3.1.7.5 REMAINDER

The contents of the NORD-50 remainder register, which is saved in block 0 variable EXT1, is printed.

3.1.7.6 OVERFLOW

The contents of the NORD-50 overflow register, which is saved in the block 0 variable EXT2, is printed (see Chapter 6).

3.1.7.7 TEST-MEMORY <address>, <number of blocks>

Tests the NORD-50 memory. The number of blocks is decimal default and block size is 1K words.

3.1.7.8 LOOP-ON

Turns on the deposit/examine loop for the TEST-MEMORY command.

3.1.7.9 LOOP-OFF

Turns off the deposit/examine loop for the TEST-MEMORY command.

3.1.7.10 MEMORY-MAP [<output file>]

The command lists a map of the NORD-50 memory. The command is especially useful for multiple NORD-50 CPUs with shared memory. In this case a cross-reference of memory addresses as seen from all the CPUs is listed. All addresses are given in octal page numbers (1 page = 512 words). To get the NORD-50 address multiply the octal page number by 1000_8. The command can only be issued by user system.

Example: (next page)

The memory map of a system with four CPUs (some manual editing has been accomplished).

*MEMORY [illegible]-MAP

MEMORY MAP OF YOUR NORD-50 SYSTEM

SEEN FROM NORD-50/1

PAGE NO.	N50/1	N50/2	N50/3	N50/4
0000	0000	N	N	N
0020	0020	N	N	N
0040	0040	N	N	N
0060	0060	N	N	N
0100	0100	0100	0100	0100
0120	0120	0120	0120	0120
0140	0140	0140	0140	0140
0160	0160	0160	0160	0160
0200	0200	N	N	N
0220	0220	N	N	N
0240	0240	N	N	N
0260	0260	N	N	N
0300	0300	N	N	N
0320	0320	N	N	N
0340	0340	N	N	N
0360	0360	N	N	N
0400	0400	N	N	N
0420	0420	N	N	N
0440	0440	N	N	N
0460	0460	N	N	N
0500	0500	0500	N	N
0520	0520	0520	N	N
0540	0540	0540	N	N
0560	0560	0560	N	N

SEEN FROM NORD-50/2

PAGE NO.	N50/1	N50/2	N50/3	N50/4
0000	N	0000	N	N
0020	N	0020	N	N
0040	N	0040	N	N
0060	N	0060	N	N
0100	0100	0100	0100	0100
0120	0120	0120	0120	0120
0140	0140	0140	0140	0140
0160	0160	0160	0160	0160
0200	N	N	N	N
0220	N	N	N	N
0240	N	N	N	N
0260	N	N	N	N
0300	N	N	N	N
0320	N	N	N	N
0340	N	N	N	N
0360	N	N	N	N
0400	N	N	N	N
0420	N	N	N	N
0440	N	N	N	N
0460	N	N	N	N
0500	0500	0500	N	N
0520	0520	0520	N	N
0540	0540	0540	N	N
0560	0560	0560	N	N

SEEN FROM NORD-50/3

PAGE NO.	N50/1	N50/2	N50/3	N50/4
0000	N	N	0000	N
0020	N	N	0020	N
0040	N	N	0040	N
0060	N	N	0060	N
0100	0100	0100	0100	0100
0120	0120	0120	0120	0120
0140	0140	0140	0140	0140
0160	0160	0160	0160	0160
0200	N	N	N	N
0220	N	N	N	N
0240	N	N	N	N
0260	N	N	N	N
0300	N	N	N	N
0320	N	N	N	N
0340	N	N	N	N
0360	N	N	N	N
0400	N	N	N	N
0420	N	N	N	N
0440	N	N	N	N
0460	N	N	N	N
0500	N	N	N	N
0520	N	N	N	N
0540	N	N	N	N
0560	N	N	N	N
0600	N	N	0600	N
0620	N	N	0620	N
0640	N	N	0640	N
0660	N	N	0660	N
0700	N	N	0700	0700
0720	N	N	0720	0720
0740	N	N	0740	0740
0760	N	N	0760	0760

SEEN FROM NORD-50/4

PAGE NO.	N50/1	N50/2	N50/3	N50/4
0000	N	N	N	0000
0020	N	N	N	N
0040	N	N	N	N
~~0060~~	~~N~~	~~N~~	~~N~~	~~N~~
0100	0100	0100	0100	0100
0120	0120	0120	0120	0120
0140	0140	0140	0140	0140
0160	0160	0160	0160	0160
0200	N	N	N	N
0220	N	N	N	N
0240	N	N	N	N
0260	N	N	N	N
0300	N	N	N	N
0320	N	N	N	N
0340	N	N	N	N
0360	N	N	N	N
0400	N	N	N	N
0420	N	N	N	N
0440	N	N	N	N
0460	N	N	N	N
0500	N	N	N	N
0520	N	N	N	N
0540	N	N	N	N
0560	N	N	N	N
0600	N	N	N	N
0620	N	N	N	N
0640	N	N	N	N
0660	N	N	N	N
0700	N	N	0700	0700
0720	N	N	0720	0720
0740	N	N	0740	0740
0760	N	N	0760	0760

3.1.8 *Commands for System Supervisor*

3.1.8.1 SET-MEMORY <N-10 page no.>, <segment no.>, <size>, <type>

This command is used to set contents in NORD-50 Monitor Segment Table. For example, when NORD-50 Monitor is implemented, the command is used to set the NORD-50 Monitor Segment Table, according to the segment creation performed in the RT loader.

Parameters:

1. NORD-10 page number for NORD-50 address 0
2. Segment number (−1 for local and core common)
3. size in pages
4. type: 0 = dynamic, 1 = static, 2 = common, 3 = local, 4 = local with DMA access and 5 = hole

Parameters 2 - 4 are repeated until segment number is zero.

The command can only be issued by user SYSTEM.

3.1.8.2 LIST-MEMORY <output file>

This command lists the segment table.

Example:

```
*LIST-MEMORY 1 ↵
SEGMENT NO.     FIRST ADDR.     SIZE     IN USE   TYPE
     061          000000        0020       X      DYNAMIC
     046          020000        0020              DYNAMIC
     047          040000        0020              DYNAMIC
     050          060000        0020              DYNAMIC
     051          100000        0020              DYNAMIC
     052          120000        0020              DYNAMIC
     053          140000        0020              DYNAMIC
     054          160000        0020              DYNAMIC
     055          200000        0020       X      DYNAMIC
     056          220000        0020       X      DYNAMIC
     057          240000        0020       X      DYNAMIC
     060          260000        0020       X      DYNAMIC
    NONE          300000        0020       X      LOCAL MEMORY
                                                  FOR NORD-50

NORD-10 PAGE NO. FOR NORD-50 ADDRESS 0:           0100
```

3.1.8.3 RESERVE-MEMORY

Static memory is reserved for NORD-50 use.

3.1.8.4 RELEASE-MEMORY

Both static and dynamic memory is released.

3.1.8.5 STOP

Stops NORD-50 if it is running.

3.1.8.6 SINTRAN or EXIT

Returns the SINTRAN III command processor and releases the NORD-50 and NORD-50 memory.

3.1.8.7 QUIT

Leave the NORD-50 Monitor and enter the SINTRAN III command processor. Memory reservation and protect setting remain unchanged.

3.1.8.8 CC<comment>

3.1.8.9 HELP <command>, [<output file>]

```
*HELP ↵
COMMAND: ↵
BREAK-ADDRESS <ADDRESS>
BREAK-CONDITIONS <BP> <BQ> <CONDITIONS>
STATUS
HELP <COMMAND> [<OUTPUT FILE>]
PRINT <LOW ADDR.> <HIGH ADDR.> [<OUTPUT FILE>]
DUMP-ADDRESS <LOW ADDR.> <HIGH ADDR.> [<OUTPUT FILE>]
OPEN-FILE <NAME> <NUMBER> <MODE>
HARDWARE-STATUS
PLACE <FILE>
LOAD–AND–GO <FILE>
SINTRAN
GOTO <ADDRESS>
RUN
MASTER-CLEAR
STOP
SAVE <FILE>
FORMAT <FORMATS>
RESET-BREAK <BREAK POINT NUMBER>
LOOK-AT <ADDRESS>
CARRY
REMAINDER
OVERFLOW
HARDWARE-GOTO <ADDRESS>
C
LOOP-ON
LOOP-OFF
RESERVE-MEMORY
RELEASE-MEMORY
CC
CPU-TIME
ZERO-MEMORY
HISTOGRAM-ON <FIRST ADDR.> <WORDS/CHANNEL> <NO. OF CHANNELS>
HISTOGRAM-OFF
HISTOGRAM-WRITE [<FILE>]
TEST-MEMORY <LOW ADDR.> <NO. OF PAGES>
ENTRIES-DEFINED [<FILE>]
ENTRIES-UNDEFINED [<FILE>]
CLOSE-FILE <FILE NUMBER>
DISPLAY <VARIABLE> [<PROGRAM UNIT>]
DUMP-VARIABLES <PROGRAM UNIT NAME> [<OUTPUT FILE>]
DUMP-ALL-VARIABLES [<OUTPUT FILE>]
LIST-UNITS [<OUTPUT FILE>]
X-FORMAT <FORMATS>
DUMP-ARRAY <ARRAY NAME> <PROGRAM UNIT> [<OUTPUT FILE>]
NEST
CHANGE <VARIABLE> <PROGRAM UNIT> <VALUE>
BREAK-NUMBER <NUMBER>
CLEAR-BREAKS
BREAK-LINE <LINE NO.> [<PROGRAM UNIT>]
LIST-BREAKS
EXHIBIT-ADDRESS <ADDRESS> <ADDRESS>
EXHIBIT-LINE <LINE NO.> <PROGRAM UNIT> <VARIABLE> [<PROGRAM UNIT>]
CONDITIONAL-BREAK-ADDRESS <ADDRESS> <ADDRESS> <VALUE> <VALUE>
CONDITIONAL-BREAK-LINE <LINE> <PROGRAM UNIT> <VARIABLE> <PROGRAM
UNIT> <VALUE>
```

```
TRAP-ADDRESS <LOW ADDRESS> <HIGH ADDRESS> <CONDITIONS>
TRAP-VARIABLE <ARIABLE> <PROGRAM UNIT> <CONDITIONS>
RESET-TRAP
EXHIBIT-OPEN <OUTPUT FILE>
EXHIBIT-CLOSE
SET-MEMORY <FIRST ADDR.> <SEGM. NO.> <SIZE> <TYPE>
EXIT
EX
QUIT
LIST-MEMORY [<OUTPUT FILE>]
LIST-TITLE
MEMORY-MAP [<OUTPUT FILE>]
```

3.1.8.10 LIST-TITLE

The NORD-50 Monitor title is listed.

Example:

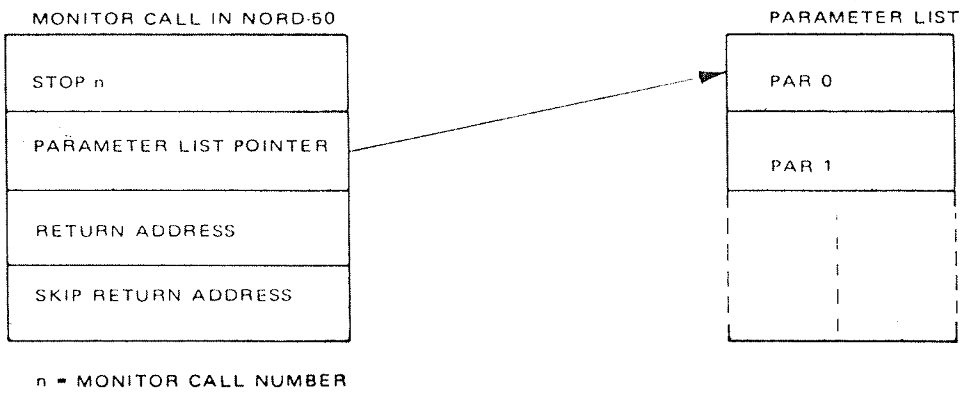
```
*LIST-TITLE ↲
NORD-50/3 MONITOR  XYZ
*
```

For multiple NORD-50 CPUs the number after the slash denotes the CPU number. The same title is listed when the NORD-50 Monitor is entered.

3.2 *MONITOR CALLS AND FORTRAN ROUTINES*

When NORD-50 programs requitre the execution of a SINTRAN III Monitor call, the Monitor call parameters fromthe NORD-50 program are transferred to the relevant NORD-10 registers by the NORD-50 Monitor before the Monitor instruction is executed.

Monitor call format in the NORD-50 program:



Appendix B lists all the SINTRAN III Monitor calls and FORTRAN routines available from NORD-50 programs.

Detailed information about SINTRAN III Monitor calls in "SINTRAN III User's Guide".

Example in NORD-50 assembler:

```
          .
          .
          .
          STOP 2                      OUTBYTE
          ACN PARAM, 0                POINTER TO PARAMETERS
          RTJ 0, ERROR                ERROR RETURN
          LDR– – – –                  OK
          .
          .
PARAM     GCN   I                     FILE NO.
          GCN   #A                    BYTE
          .
          .
```

3.3 DIRECT FILE TRANSFER

3.3.1 Direct File Transfer with RFILE and WFILE (Disk)

Direct file transfer is a feature for optimized disk transfer to NORD-50.

The file is opened by the OPEN-FILE command, modus D or DC or from the NORD-50 program by the monitor call OPEN, modus 8 or 9.

In modus 8, the file is kept open; in modus 9 the file is closed during the file transfer.

The modus 9 feature allows the user to work on a larger number of files than the maximum number of files that can be concurrently open in the SINTRAN III system.

The standard calls RFILE, WFILE and WAITF are used in the NORD-50 program, but there are some limitations to the argument.

— The actual file transfer is performed by the monitor call ABSTR. The File System is bypassed and the mass storage device may be used in an optimal way.

— The monitor calls RMAX and SMAX may be used if the file is opened (modus 8).

The limitations versus using the standard SINTRAN III file system are:

— The file must be continuous.

— Only the monitor calls OPEN, CLOSE, RFILE, WFILE, WAITF, RMAX and SMAX may be used.

— The logical block size is always 1/4 pages (128 NORD-50 words).

— The word count in RFILE or WFILE should be a multiple of the hardware sector size of the mass storage device.

— The file system object entry for the file is not updated by RFILE and WFILE.

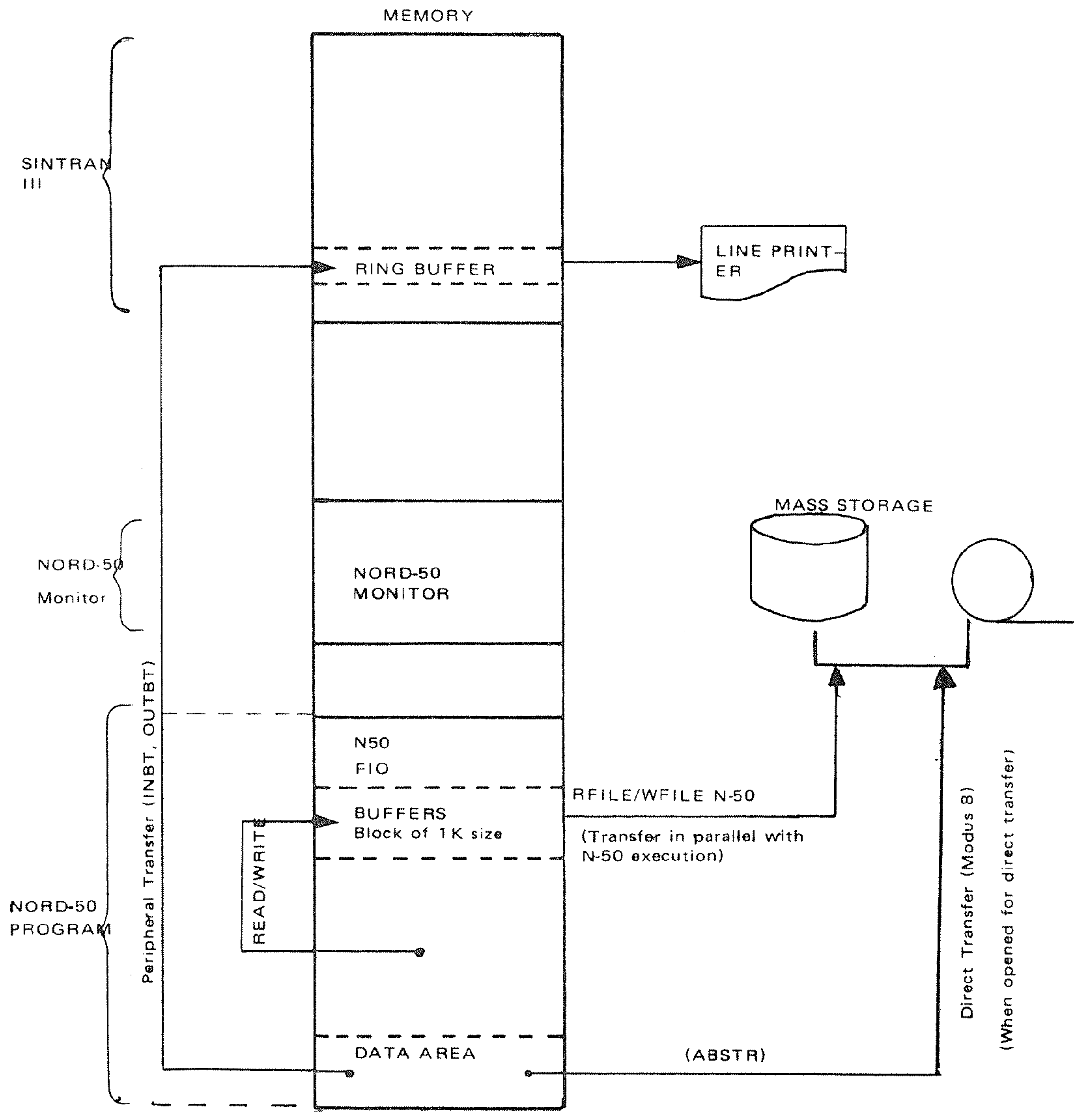
— Note that when the hardware sector size is greater than 128 words some numbers cannot be used as block address. On the 33/66/75/288 Mbytes disks the hardware sector size is 256 words, thus only even numbers may be used as block addresses.

3.3 2 *Direct File Transfer with MAGTP (Magnetic Tape)*

Direct file transfer is a feature for optimized magnetic tape transfer to NORD-50.

The file is opened by the OPEN-FILE command, modus "D", or from the NORD-50 program by the monitor call OPEN, modus 8. The modus "DC" (or 9) may not be used for MAGTP.

The monitor call MAGTP may be used in a standard way from NORD-50, but the actual transfer is performed by the ABSTR monitor call in NORD-10 and it goes directly from the interface into the NORD-50 memory via the DMA channel.



3.4 *POWER FAIL*

If a program where the symbol PWF. is defined is running in the NORD-50 and the NORD-10 gets a power fail, the NORD-50 is stopped by the NORD-10. PC is saved in memory and the NORD-50 is started in the address of PWF.. On power up the NORD-50 is started in the address PWF. + 1. A standard register save/unsave sequence with the name PWF. is included on the N50-FIO.

4 SYSTEM ARCHITECTURE

4.1 *SYSTEM PARTS*

NORD-50 Monitor consists of two parts:

— one core resident part which is a part of SINTRAN III (specified when SINTRAN III is generated)

— a part on a separate segment (loaded by the RT loader)

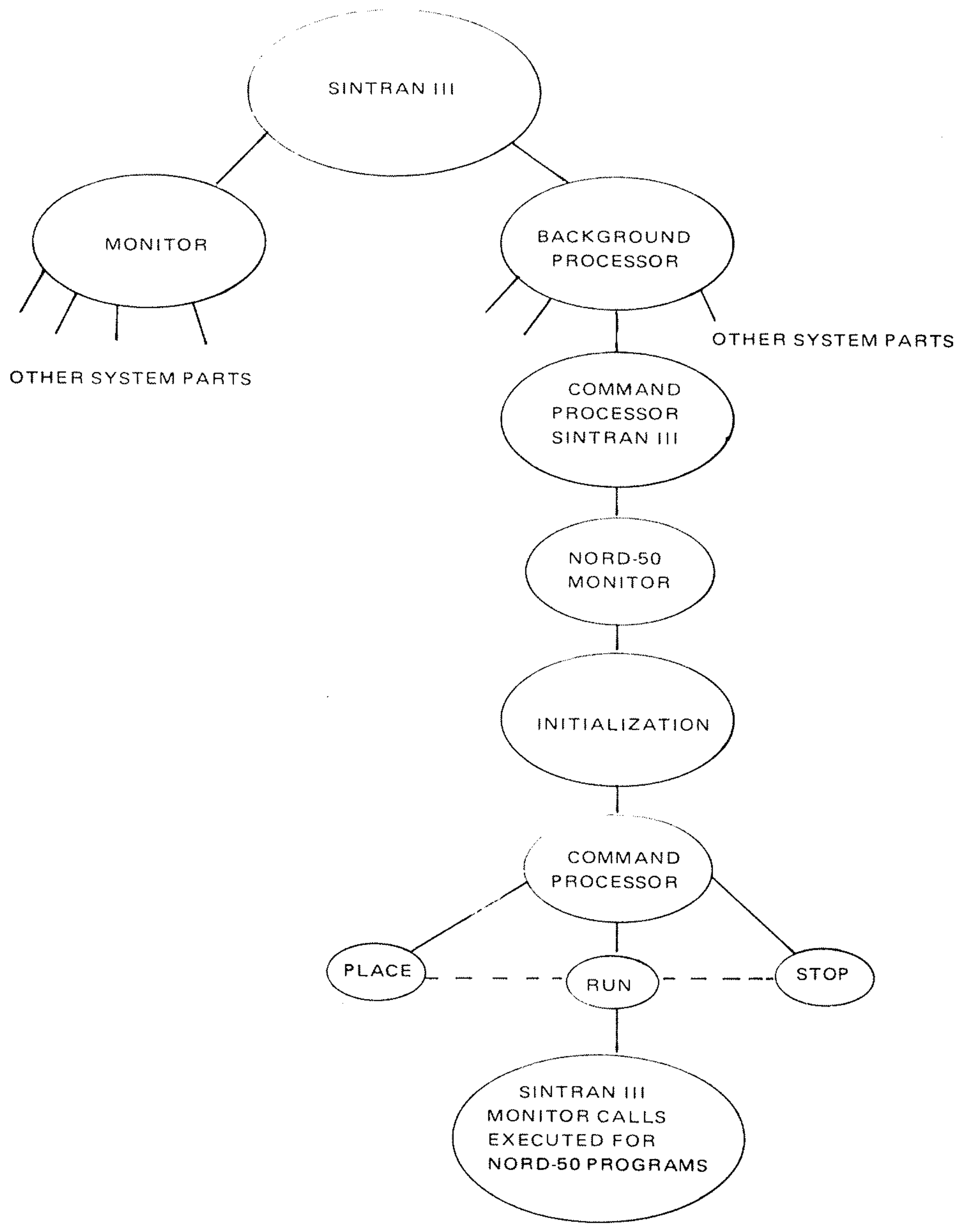


Figure 4.1: NORD-50 Monitor System Parts

4.2 *MEMORY LAYOUT (Typical Layout)*

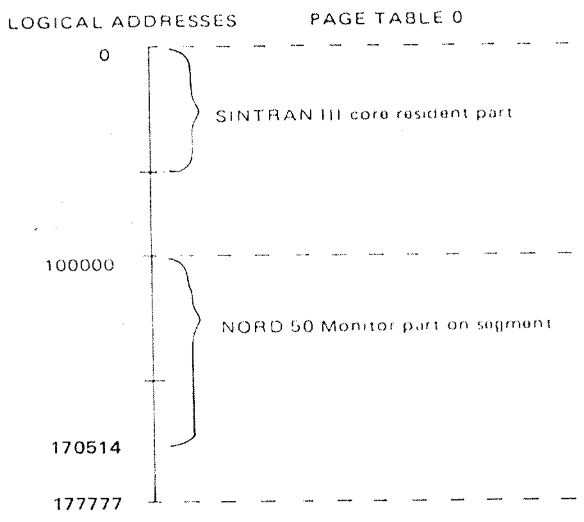

LOGICAL ADDRESSES
PAGE TABLE 0
0
SINTRAN III core resident part
100000
NORD 50 Monitor part on segment
170514
177777

Figure 4.2: Example of Memory Layout for a NORD-10/NORD-50 Configuration

The core resident part of the NORD-50 Monitor is placed somewhere in the SINTRAN III paging area and consists of:

- NORD-50 Data field
- Enter NORD-50 Monitor routine (called from SINTRAN III background processor) (Hardware level 1)
- Interrupt drivers (hardware levels 3 and 12)
- subroutines for READ-FILE/WRITE-FILE
- subroutines ABSTR (for direct file access)
- subroutine for MAGTP monitor call

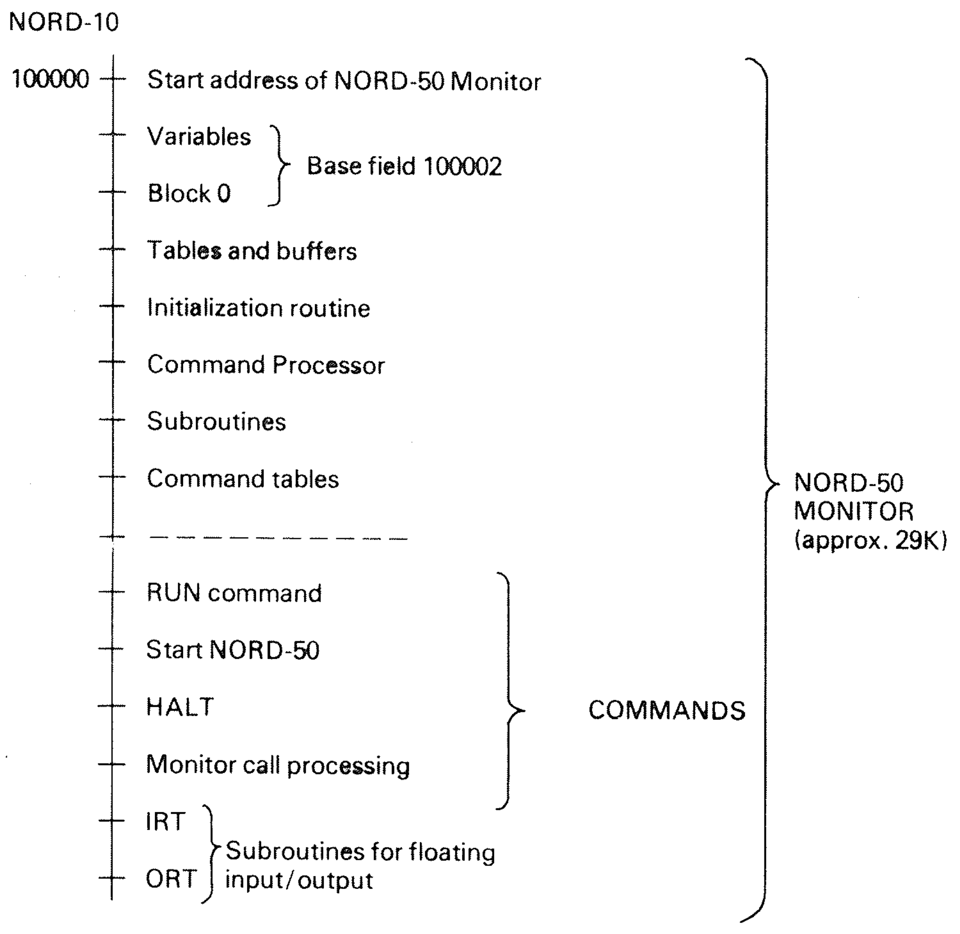


Figure 4.3: Memory Layout of NORD-50 Monitor Segment Part

4.3 *USE OF MEMORY MANAGEMENT IN NORD-50 MONITOR*

The NORD-50 may share memory with other processors like the NORD-10. The NORD-50 Monitor assumes that the NORD-50 shares some or all of its memory with a processoor, supervises the execution of the NORD-50 programs and determines which parts of memory may be used by the NORD-50 or by the SINTRAN III processor.

To address the NORD-50 memory from the SINTRAN III processor a number of segments are used. Descriptions of those segments are put in a table in the NORD-50 Monitor, the NORD-50 memory segment table, at system implementation (see Chapter 5).

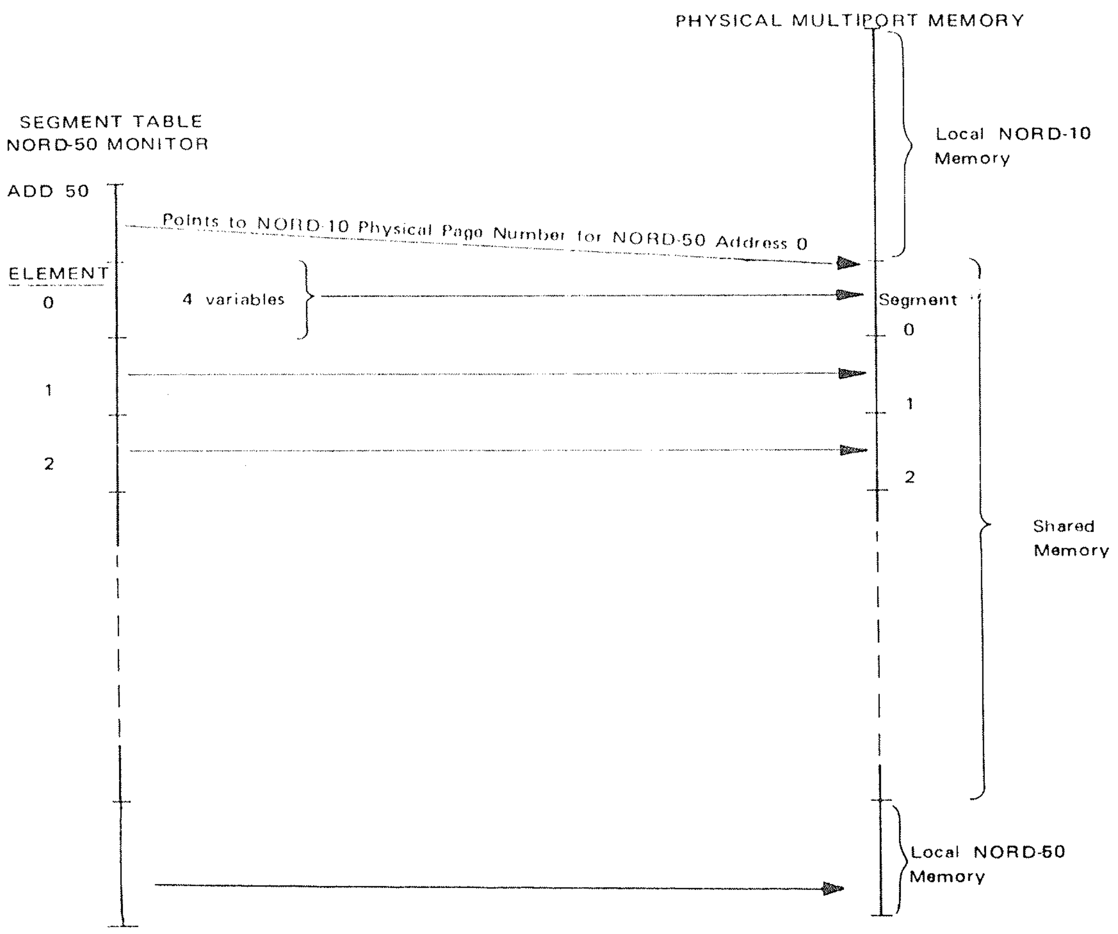


Figure 4.4: The segment table contains one element of 4 variables for each segment in shared memory. The segment elements contain the necessary information about the segments.

The contents of the segment table is set by the SET-MEMORY command.

When some part of memory is allocated for NORD-50, the segment table is used to place the segments in the right place in Physical Memory. That is, the same segment is always placed in the same physical address in Memory. (This is done when using the NORD-50 Monitor commands BREAK-CONDITIONS, PLACE, LOAD and RESERVE and they use the SINTRAN III Monitor call FIXC – see below.)

The Memory may be divided into 6 groups;

— Dynamic Memory
— Static Memory
— RT Common Memory
— Local NORD-50 Memory
— Local NORD-50 Memory with DMA access
— Hole

according to their allocation strategies.

1. *Shared Memory Part*

   Shared memory is the part of memory which may be used both by the NORD-50 or the SINTRAN III processor. When the NORD-50 is not in use, all the shared memory may be used by the SINTRAN III processor.

   The programmer controls allocations of shared memory by setting the NORD-50 address violation system:

   a) If the break conditions any reference and store reference are *not* on or BP > BQ all the NORD-50 memory is reserved for NORD-50.

   b) If the break condition any reference or store reference is on and BP ⩽ BQ the address intervals [0, BP > and [BQ, MAX] are reserved for the NORD-50 and the area [BP, BQ > is reserved for the SINTRAN III processor. MAX is the upper address in the NORD-50 memory. BP is the upper address in the NORD-50 memory segment which contains the address in breakpoint register 1 (BP). BQ is the lower address in the segment which contains the address in breakpoint register 2 (BQ).

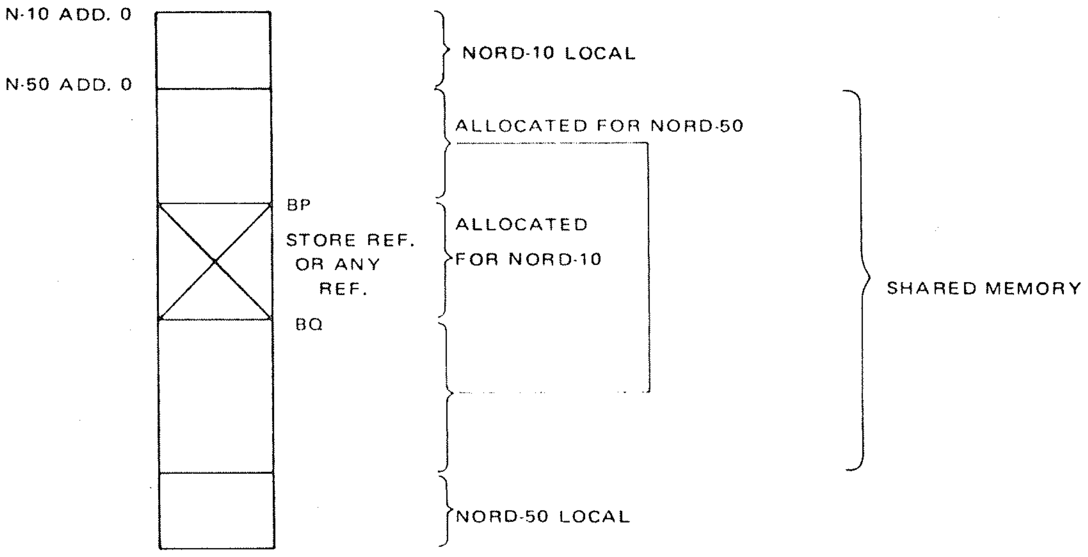


Figure 4.5: An Example of Allocated Shared Memory

Memory is allocated for the NORD-50 with the monitor call FIXC in the SINTRAN III processor (FIXC is called from the BREAK-CONDITION command in the NORD-50 monitor) and is released for use by the SINTRAN III processor with the monitor call UNFIX. (The FIXC command is used to set a continuous area because the NORD-50 has no paging hardware.)

Shared memory may be divided into two groups:

i) Dynamic Memory

Some of the segments may be declared as dynamic. Those segments are allocated and released any time the available memory area for NORD-50 is changed.

ii) Static Memory

Some of the segment may be declared as static. Those segments are allocated for NORD-50 the first time they are used after a SINTRAN III cold start (Master Clear, Load) or after a RELEASE-MEMORY command in the NORD-50 Monitor. However, they are not released when new programs which do not use them run on the NORD-50 or when the NORD-50 is not in use.

2. *RT Common*

If the SINTRAN III processor's RT common is a part of the NORD-50 memory, it may be used by both processors simultaneously.

The size and location of the RT common area must be established at SINTRAN III generation time or by use of the SINTRAN-SERVICE program.

3. *Local NORD-50 Memory*

Local NORD-50 memory is a part which is not available for the SINTRAN III processor. Local memory may be used in a NORD-10/NORD-50 system with more than 256K words or in any system where sharing of some part of memory is not desired.

In the NORD-50 Monitor accessing of NORD-50 memory is always performed by using the NORD-50 hardware register (EXAMINE/DEPOSIT), even when shared memory is used. (For example, when fetching monitor call parameters, transfer of data from NORD-50 Monitor buffer to NORD-50 Memory, etc.)

5 IMPLEMENTATION OF NORD-50 MONITOR

5.1 *SINTRAN III GENERATION*

SINTRAN III should be generated with NORD-50 drivers, library symbol "8N50".

The SINTRAN III variable FIXMAX (SINTRAN III listing — Section 3.4.1) should have a value at least equal to the number of memory pages shared by NORD-10 and NORD-50 (shared memory — refer to Section 4.3).

This chapter describes the procedures to:

— load the NORD-50 Monitor

— declare and allocate necessary segments under SINTRAN III (by the RT Loader)

— declare and allocate the memory space to be used by the NORD-50 (the SET-MEMORY command in the NORD-50 Monitor)

These tasks must be done by the user SYSTEM.

5.2 LOADING OF THE NORD-50 MONITOR ON SEGMENT

The NORD-50 Monitor is supplied on Floppy.

Loading Procedure: (Output from the computer is removed)

The NORD-50 MONITOR can be loaded non-reentrant or reentrant. Only users with a multiple NORD-50 system shall use the reentrant version.

Not-reentrant load procedure:

```
@RT-L
CL-S 12
S-P-T 0
N-S 12 2 DM FRW,,
Y
A-A 12 72000 100000
Y
END
READ-BIN N50MON:BPUN,12
YES
END
EXIT
```

Reentrant load procedure:

```
@RT-L
CL-S 12
Y
CL-S 15
Y
CL-S 16
Y
CL-S 17
Y
CL-S 20
Y
S-P-T 0
N-S 12 2 DM FR,,
Y
A-A 12 60000 112000
Y
END
S-P-T 0
N-S 15 2 DM WRF,,
Y
A-A 15 12000 100000
Y
END
S-P-T 0
N-S 16 2 DM WRF,,
Y
A-A 16 12000 100000
Y
END
READ-BIN N50MON-2:BPUN,12
YES
END
READ-BIN N50MON-1:BPUN 15
YES
END
READ-BIN N50MON-1:BPUN 16
Y
END
READ-BIN N50MON-1:BPUN 16
Y
END
READ-BIN N50MON-1:BPUN 17
Y
END
READ-BIN N50MON-1:BPUN 20
Y
END
EXIT
```

5.3 *CREATING NORD-50 SEGMENTS UNDER SINTRAN III*

The NORD-50 memory segments should be:

- on ring 0, 1 or 2
- non-demand
- read and write permitted
- written in page bit set
- on page table 1, 2 or 3

The logical address space should be:

- from address 0.
- limited upwards by the upper free address on the paging table.

(See also Section 4.3.)

Example:

Two segments with 16 and 8 NORD-10 pages will be created in this example.

```
@RT-LOADER ↵
REAL-TIME LOADER      78.09.05
*CLEAR-SEGMENT 46 ↵
*NEW-SEGMENT 46   2   ND   RW   WP ↵
*SET-PAGE-TABLE 1 ↵
*ALLOCATE-AREA  ↵
SEGMENT NO.: 46 ↵
AREA SIZE: 40000 ↵
LOWER ADDRESS: 0 ↵
```

That is, segment number 46 is defined with:

- area size (logical address space) of 40000_8 NORD-10 words = 20_8 pages = 16_{10} pages
- lower address (the first logical address of the area) is set to 0.

```
*END-LOAD ↵
*CLEAR-SEGMENT 47 ↵
*NEW-SEGMENT 47   2   ND   RW   WP ↵
*SET-PAGE-TABLE 1 ↵
*ALLOCATE-AREA 47   20000   0  ↵
*END ↵
*EXIT ↵
```

Segment no. 47 is defined with 20000_8 NORD-10 words $= 10_8 = 8_{10}$ pages

5.4 *ALLOCATING MEMORY SPACE FOR NORD-50*

When the NORD-50 Monitor is implemented and the NORD-50 segments are created under SINTRAN III, the NORD-50 Monitor segment table must be set according to the segment creation performed in the RT loader (see Section 5.3). This is done by the NORD-50 Monitor command SET-MEMORY (see also Figure 4.4). (The parameter list for the command is described in Section 3.1.6.)

Defining the NORD-50 configuration according to the example in Section 5.3:

```
@NORD-50 ↵
*SET-MEMORY ↵
NORD-10 PAGE NO.: 100 ↵
SEGMENT NO.: 46 ↵
SIZE: 20 ↵
TYPE: 0 ↵
```

That is:

- NORD-10 physical page number for NORD-50 address 0 is 100_8 according to the physical connection (see Figure 5.1)
- segment number is 46
- segment size 20_8 pages (see Section 4.3)
- segment type is dynamic

This information is set into the segment table in the NORD-50 Monitor (see Figure 3.1).

```
SEGMENT NO.: 47 ↵
SIZE: 10 ↵
TYPE: 1 ↵                    (static)

SEGMENT N.: -1 ↵             (local and RT common)
SIZE: 20 ↵
TYPE: 3 ↵                    (local)

SEGMENT NO.: 0               (parameters are no longer repeated)
```

In this example the first NORD-50 page in NORD-10 is 64_{10}.

The NORD-50 Memory consists of:

- segment no. 46_8 (16_{10} pages, dynamic)
- segment no. 47_8 (8_{10} pages, static)
- 16_{10} pages local NORD-50 memory

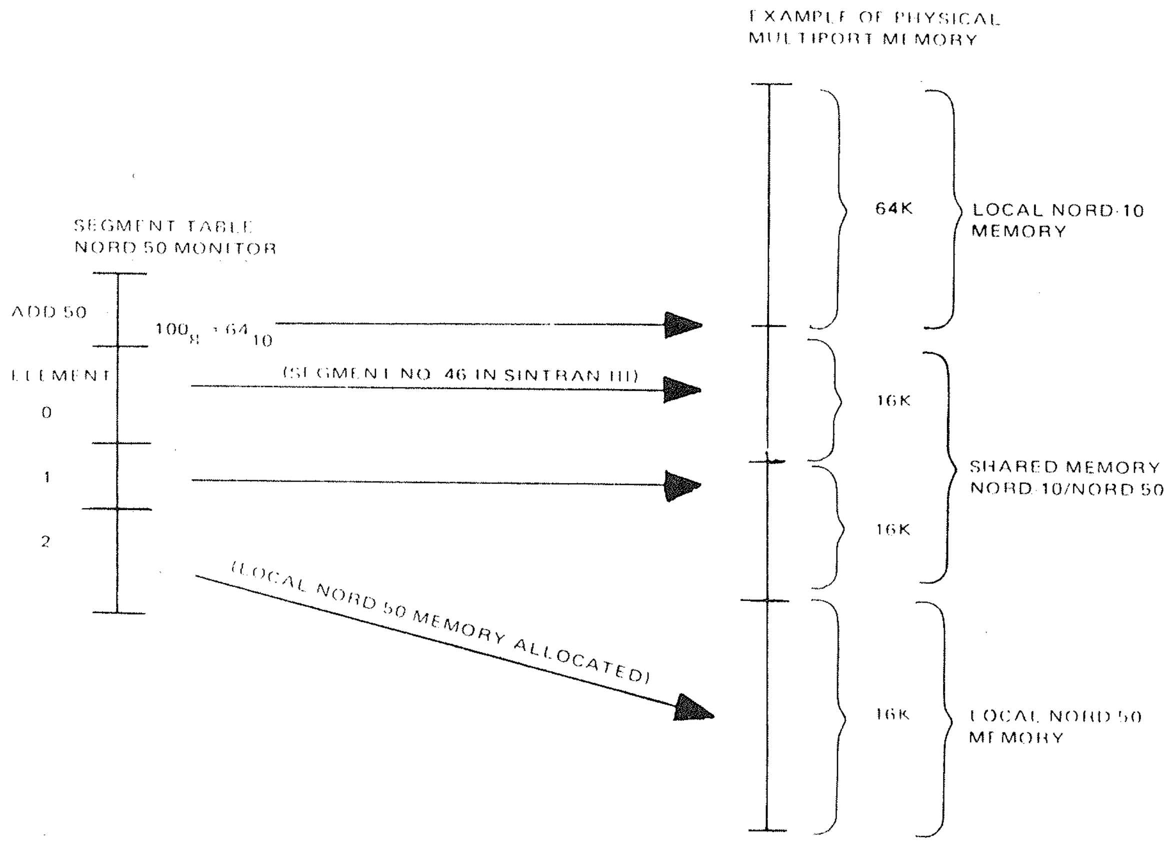

EXAMPLE OF PHYSICAL MULTIPORT MEMORY
SEGMENT TABLE NORD 50 MONITOR
ADD 50
100₈ = 64₁₀
ELEMENT
0
1
2
(SEGMENT NO 46 IN SINTRAN III)
(LOCAL NORD 50 MEMORY ALLOCATED)
64K
LOCAL NORD-10 MEMORY
16K
16K
SHARED MEMORY NORD-10/NORD-50
16K
LOCAL NORD-50 MEMORY

Figure 5.1: The memory allocation according to the example above (1K = 1024 16 bit words). Location ADD 50 in the segment table contains the NORD-10 physical page number (64_{10} in the above example) for the NORD-50 address 0.

5.5 *AN EXAMPLE OF IMPLEMENTATION*

This section gives an example of implementation of the NORD-50 Monitor with the hardware environment shown in Figure 1.2.

This is a configuration with:

— 256K words NORD-10 (16 bits). Physical memory (banks 0, 1, 2 and 3) where all the NORD-10 address space is used.

— 192K words NORD-50 (32 bits). Physical memory (banks 2, 3, 4, 5, 6 and 7) where all the NORD-50 address space is used.

— 128K NORD-10 words = 64K NORD-50 words. Physical memory is used as NORD-10/NORD-50 shared memory (banks 2 and 3).

In this example:

- NORD-50 Monitor is loaded into segment number 12.
- 8 NORD-50 segments of 16_{10} pages are created under SINTRAN III.
- NORD-50 memory is allocated for 8_{10} segments of 16_{10} pages shared memory, dynamic and 1 segment of 256 pages local memory.

```
@RT-LOADER ↵
REAL-TIME LOADER
*READ-BINARY TAPE-READER 12 ↵               LOADING OF NORD-50
CHANGING EXISTING SEGMENT?: YES ↵           MONITOR
*END ↵

*CLEAR-SEGMENT 46 ↵                         CREATING NORD-50
*NEW-SEGMENT 46  2  ND   RW   WP ↵          SEGMENTS UNDER
*SET-PAGE-TABLE 1 ↵                         SINTRAN III
*ALLOCATE-AREA 46   40000  0 ↵
*END ↵
* ↵

*CLEAR-SEGMENT 47 ↵
*NEW-SEGMENT 47  2  ND   RW   WP ↵
*SET-PAGE-TABLE 1 ↵
*ALLOCATE-AREA 47   40000  0 ↵
*END ↵
  ↵
*CLEAR-SEGMENT 50 ↵
*NEW-SEGMENT 50  2  ND   RW   WP ↵
*SET-PAGE-TABLE 1 ↵
*ALLOCATE-AREA 50   40000  0 ↵
*END ↵

*CLEAR-SEGMENT 51 ↵
*NEW-SEGMENT 51  2  ND   RW   WP ↵
*SET-PAGE-TABLE 1 ↵
*ALLOCATE-AREA 51   40000  0 ↵
*END ↵
```

```
*CLEAR-SEGMENT 52
*NEW-SEGMENT 52   2   ND   RW   WP
*SET-PAGE-TABLE 1
*ALLOCATION AREA 52   40000   0
*END

*CLEAR-SEGMENT 53
*NEW-SEGMENT 53   2   ND   RW   WP
*SET-PAGE-TABLE 1
*ALLOCATE-AREA 53   40000   0
*END-LOAD

*CLEAR-SEGMENT 54
*NEW-SEGMENT 54   2   ND   RW   WP
*SET-PAGE-TABLE 1
*ALLOCATE-AREA 54   40000   0
*END

*CLEAR-SEGMENT 55
*NEW SEGMENT 55   2   ND   RW   WP
*SET-PAGE-TABLE 1
*ALLOCATE-AREA 55   40000   0
*END
*EXIT

@NORD-50                                ALLOCATING MEMORY
*SET-MEMORY                             SPACE FOR NORD-50
NORD-10 PAGE NO.: 200
SEGMENT NO.: 46
SIZE: 20
TYPE: 0
SEGMENT NO.: 47
SIZE: 20
TYPE: 0
SEGMENT NO.: 50
SIZE: 20
TYPE: 0
SEGMENT NO..: 51
SIZE: 20
TYPE: 0
SEGMENT NO.: 52
SIZE: 20
TYPE: 0
SEGMENT NO.: 53
SIZE: 20
TYPE: 0
SEGMENT NO.: 54
SIZE: 20
TYPE: 0
SEGMENT NO.: 55
SIZE: 20
TYPE: 0
SEGMENT NO.: –1
SIZE: 400
TYPE: 3
SEGMENT NO.: 0
*EXIT
@
```

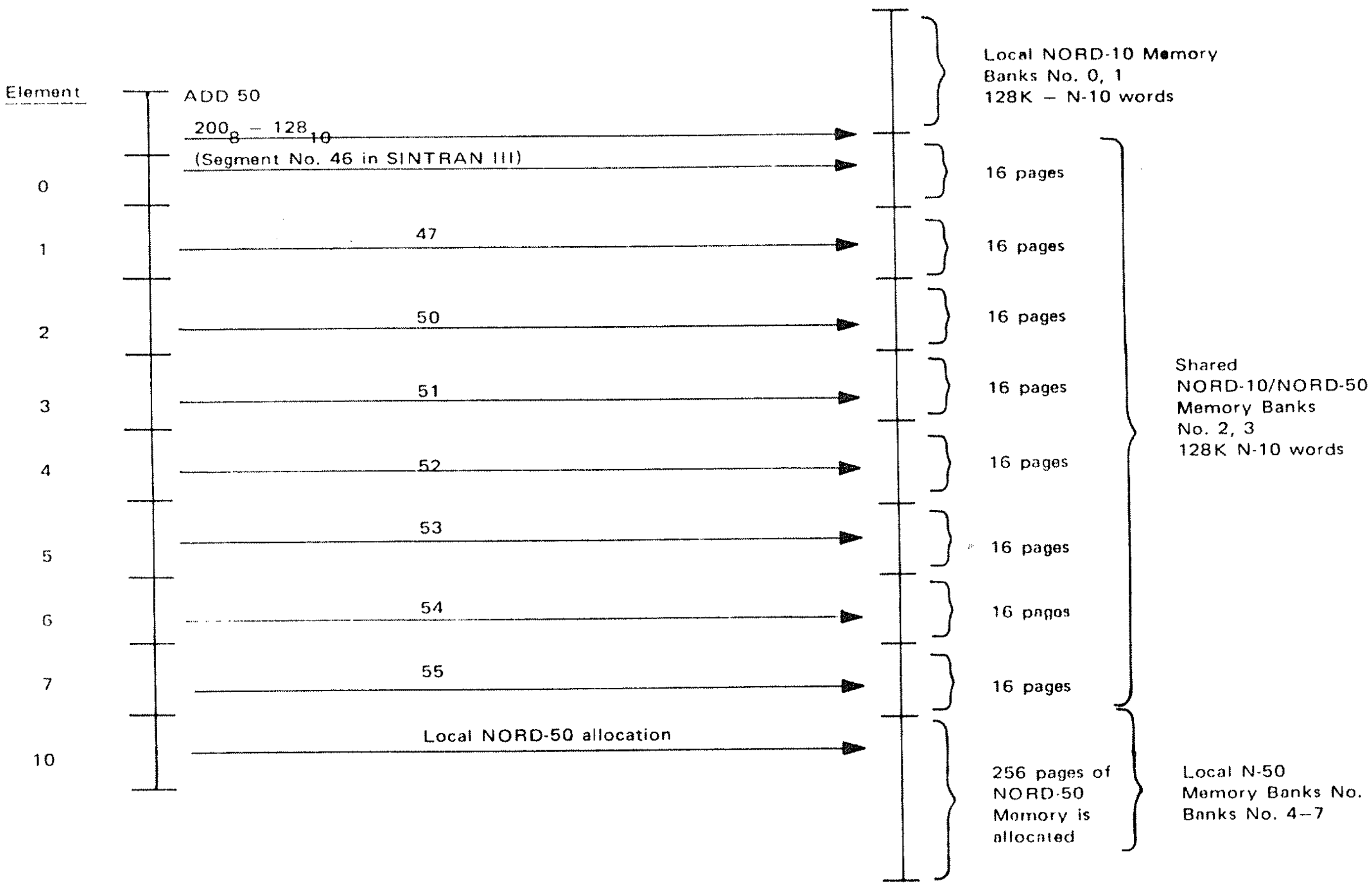

Element
ADD 50
$200_8 - 128_{10}$
(Segment No. 46 in SINTRAN III)
0
1
47
2
50
3
51
4
52
5
53
6
54
7
55
10
Local NORD-50 allocation
Local NORD-10 Memory Banks No. 0, 1 128K – N-10 words
16 pages
16 pages
16 pages
16 pages
16 pages
16 pages
16 pages
16 pages
Shared NORD-10/NORD-50 Memory Banks No. 2, 3 128K N-10 words
256 pages of NORD-50 Memory is allocated
Local N-50 Memory Banks No. Banks No. 4–7

Figure 5.2: The memory allocation according to the example above. Location ADD50 in the segment table contains the NORD-10 physical page number (128_{10} in the above example) for NORD-50 address 0.

6 NORD-50 MONITOR DOCUMENTATION

6.1 *GENERAL DESCRIPTION*

The monitor works on hardware level 1 in NORD-10 under control of SINTRAN III background command processor, regarded as a system segment. It is run on ring 2 and uses page index table 0.

By the NORD-50 monitor, several tasks can be performed. For instance:

— Take a program in executable format, prepared by the NORD-50 loader on a file and place it in the NORD-50 memory (PLACE command).

— Open files and connect them to file numbers used by the program (OPEN-FILE command).

— Start execution of a program in NORD-50 (RUN command).

— Hardware and software debugging facilities (BREAKPOINT, LOOK-AT, TEST-MEMORY, etc.).

— System supervisory commands (SET-MEMORY, BREAK-CONDITIONS, etc.)

This chapter describes the different parts of the Monitor according to the listing and includes:

— NORD-50 Monitor Data Part (Section 6.2)

— NORD-50 Monitor Initialization Part (Section 6.3)

— NORD-50 Monitor Command Processor (Section 6.4)

— NORD-50 Monitor Command Routines (Section 6.5)

The RUN routine (Section 6.5) is the most important part for understanding the function of the NORD-50 Monitor.

6.2 *DATA DEFINITIONS*

Base field	Start address
	Subroutine WORKING SPACE
	Register dump area (NORD-10 registers (used if fail situation)
	STACK-POINTER (STPNT)
	Device number for current terminal (TCDEV)
	SUBROUTINE entry points
	Program variables
	Area for parameter transfer between N-50 and N-10 (by SINTRAN III monitor calls)
	SA TA TD } Hardware register saved at last stop
	Block 0 (area which contains register and status information for the current program in NORD-50)
	Saved copy of block 0
	Monitor call parameter pointer list
	File transfer buffer (1 page) (used, for example, for file transfer to local memory in N-50)
	Connected file no. conversion table (128_{10} word), Index connected file number in N-50 program. Content = Real file no. in N-10.
	Histogram table
	STACK
	Open file name string buffer
	Parameter string buffer
	Command string buffer
	Debug file name string buffer
	NORD-50 memory segment table
	Direct file table
	Breakpoint table
	Debug unit table
	Routine to execute monitor calls
	Debug variable text buffer
	Table of no. of parameters in N-50 monitor calls
	Table of start address of individual monitor call processing routines

Core Resident Part

NORD-50 Data field: (refer to page 4–2)

Word No.	*Symbol*	*Contents*	*Explanation*
–6		50TMR	
–5		0	Standard
–4		0	SINTRAN III
–3	DIOX	IOX 30	NORD-50 Device
–2	CIOX	IOX 60	NORD-50 Control
–1		RSIN5	
0		0	
1		0	Standard SINTRAN III
2		*–2	
3		0	
4			0
5		0	
6		50DM0	Routine monitor level
7	5ORFILE	50RF1	READ FILE ROUTINE FOR N-50
10	5OWFILE	50WF1	WRITE FILE ROUTINE FOR N-50
11	50TID	0	Value of ATIME when N-50 has
12		0	stopped
13	ATID	ATIME	Address of ATIME
14	50CF	1	Cold start flag (master clear load 1 when N-50 has not been started)
15	50DI	50DI1	Direct File transfer routine
16	50MT	50MT1	MAGTP routine
17	50P1	*4	Parameter list for ABSTR
20	50P2	*4	
21	50P3	*5	
22	50P4	*5	
23	50FNC	0	
24	50PHAD	0	Parameter list for ABSTR
25		0	
26	50MASAD	0	
27	50NBLCK	0	
30	50LLGPH	LOGPH	SINTRAN system routine
31	50version	5	Monitor version
32	50LDPIOF	LDPIO	SINTRAN system routine
33	N50NO	0	NORD-50 NO. if more than 1
34	50PBP	0	BP for power fail
35		0	
36	50PBQ	0	BQ for power fail
37		0	

Word No.	*Symbol*	*Contents*	*Explanation*
40	50PBL	0	Break conditions (modus) for P.F.
41	50FLA	0	Flag word
42	50HQU	0	Head of task queue
43	50SWI	0	Task
44		0	
45	50PWF	0	Entry
46		0	
47	50PC	0	Pointer
50		0	
51	50RUN	0	Running task
52	50TABU	50TSB	Task description table
53	55FUN	0	Parameters to level 12
54	50TSK	0	
55	LL12L	L50CH	Level 12 entry for task functions
56	L50SL	SLV12	SINTRAN system routine
57	N5CCTAB	CCTAB	Start of SINTRAN III RT Common
60	PUT1L	PUT1L	SINTRAN III system routine
61	GET1L	GET1L	SINTRAN III system routine
62		0	
63		0	Not used
64		0	
65		0	
66		0	
67		0	
70		0	

This data field is placed somewhere in SINTRAN III.

6.2.1 *Program Variable Definitions*

Refer to page 6–2.

Symbol	*Contents*	*Explanation*
RETADR	–	Return address to SINTRAN III
RFIL	–	RFILE routines in resident
WFIL	–	WFILE routines in resident
CHECKF	1	Check exact memory configuration if 1
DATF	–	NORD-50 Data field address
BLMAX	0	Maximum memory address in pages for current NORD-50 program (from loader)
CBMAX	0	Maximum physical page number for NORD-50
CMAX	0	Upper existing address
RTCOM	0	First address in RT common (SET-MEMORY command)
IPFIL	0	Used as file number (PLACE/LOAD)
FORMA	5	Output format (FORMAT/PRINT/LOOK-AT)
SLASH	0	Flags for command
IASK	1	processor
COMSP	0	Command string pointer
UTIME	0	Pointer to ATIME
S3SEGST	SSEGSTART	SINTRAN III segment table
TSTART	–	Time when NORD-50 was started
CPT	0	CPU time used by NORD-50 (zeroed when NORD-50 monitor is started)
HFLAG	0	Histogram on (program counter statistics)
HSTART	0	
CHANW	0	Histogram parameters
NCHAN	0	
INCHNT	0	Counts in Table (in histogram)
TOTCNT	0	Total counts (in histogram)
MEMOK	0	One if memory segment table okay
BRPCU	0	Current break entry

Symbol	Contents	Explanation
FLUT 1	0	
FLUT 2	0	
FLUT 3	0	Temporary storage
FLUT 4	0	
FLUT 5	0	
PPARAS	PARAS	Pointer to parameter string for command processor in NORD-50 monitor
NULL	0	Zero
EN	1	ONE
MINEN	−1	Minus one
BLCNT	0	Block count for place save (page counter)
MLOOP	0	Loop flag for test core common
HGOFL	0	= 1 if hardware GOTO, SAVEUNSAVE not used
ENDF	0	End flag (end of program – MON 134)
BRKMF	0	MAC break mode flag
SEGM	0	Parameters for
MEMR	0	FIX and UNFIX
CURRSEG	0	NORD-50 memory segment
PTN0	0	Page table for current segment
SEGSTART	0	First logical address of NORD-50 memory segment for NORD-10
		Parameter list for different monitor calls
IPC	0	Program counter saved for monitor call processing
PARFL	0	Parallel operation flag (if NORD-10 and NORD-50
XSTRT	0	are running at the same time)
OPMODE	0	Mode in last open
BTFLG	0	Batch mode flag
FIRST	1	First time used flag
SUFLAG		1 = UNSAVE registers lost 2 = SAVE registers lost
RUNNFL	0	Program active flag
SPECMT	0	Specify memory configuration flag

Symbol	*Contents*	*Explanation*
IPBYT	0	Pointer for N-50 console input
DIOX	0	NORD-50 DEVICE IOX instruction
CIOX	0	NORD-50 CONTROL IOX instruction
MAXT	0	Maximum CPU time in minutes
LPARAM	0	PARANT. level counter
SUPRF	0	Supress internal error messages flag
PASS	0	2 = System User, 1 = RT, 0 = Other
UNAME	0	UNIT program name
UBLIN	0	LINE number of base
UVBAS	0	Variable base of unit
UWN	0	No. of words in unit table
UWS	0	No. of words in symbol table
UBP	0	Byte pointer to current unit
SYMBP	0	Byte pointer to current unit
LINBP	0	Byte pointer line no. table of current unit
SNAME	0	Save UNAME
EXFRORM	0	Extra output formats for variables
ELEMFL	0	Array element specified flag
ELEM1	0	Parameter variable element
INDXES	0	Array element index table
EXFIL	1	
PWFL	0	Power fail flag
ANY DMA	0	Any DMA access flag

6.2.2 *Block 0 Definition*

(Refer to page 6—2.)

Block 0 is an area which contains registers and status information for the current program in NORD-50. This is the block of a NORD-50 executable format program file as prepared by the NORD-50 loader. (Read into the area by the PLACE/LOAD commands.)

Symbol	*Contents*	*Explanation*
MAINSA		Main program start address
PC		Program counter for the program
BP BQ		Breakpoint registers
DEFBP DEFBQ		Not used
STATU	204	Modus register (parity, store trap)
STN05		Status register
5CMAX		Maximum NORD-50 memory address used by the program
ADSAVE	120	Start address of save routine
ADUNSA	137	Start address of unsave routine
EXT 1 EXT 2 CARRY REGAR		Remainder Overflow used by save and Carry unsave routines Register 0 - 63
SUB- PROGRAM		Save/unsave program. A program in NORD-50 assembler code.
BITMAP	—	Bit map table for the program file

6.2.3 *NORD-50 Memory Segment Table Definition*

(Refer to page 6–2.)

The segment table is a description of the NORD-50 memory.

ADD 50	NORD-10 address of NORD-50 address 0 (page no.)

SEGMENT TABLE (SEGTAB) has 4 variables for each table entry:

Symbol	*Explanation*
SEGNO	Segment no. –1 denotes no segment for this part of memory. 0 denotes end of table.
SMEMA	Page no. in NORD-50 for start of segment
SSIZE	Segment size in pages
SFLAG	Flag bits: Bit 0; segment FIXC (in use by N-50) Bit 1; STATIC segment Bit 2; RT common Bit 3-4; paging table no. of the segment Bit 5; DMA access to this segment area

The segment table is set by the SET-MEMORY command.

6.2.4 *Direct File Table*

(Refer to page 6–2.)

Direct file table is used for optimized disk transfer for NORD-50. RFILE/WFILE calls in NORD-50 may be translated to ABSTR call by use of the "Direct File Table" information.

The table is set by the OPEN-FILE command in the NORD-50 monitor or when the monitor call OPEN is executed in NORD-50 programs.

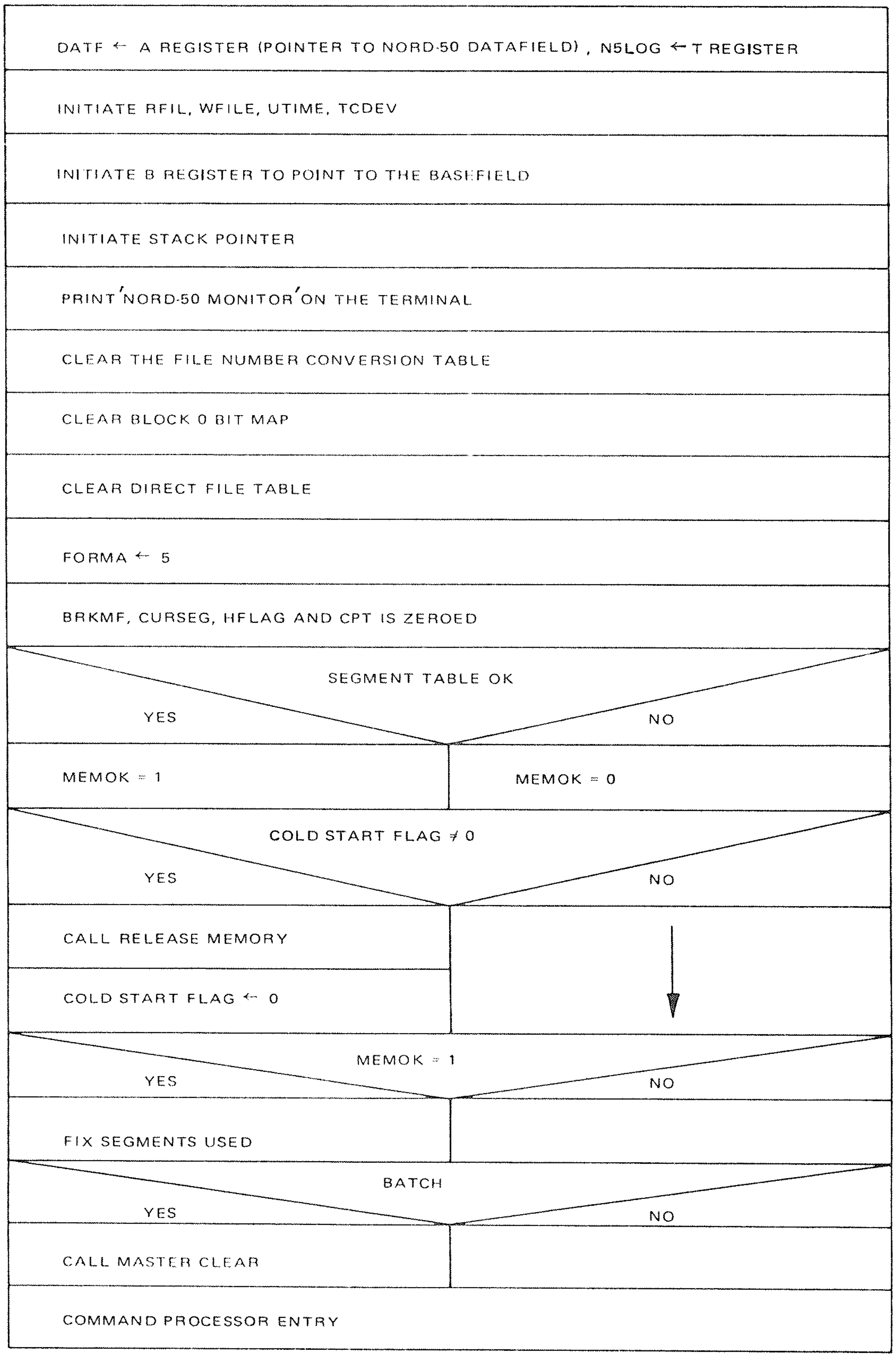
Access modus is "D" or "DC" for the OPEN command and 8 or 9 for the monitor call.

There are 6 variables in the table foor each file opened for direct file transfer:

Symbol	*Explanation*
DLOG	Logical no. for the mass storage device
DUNIT	Device unit no.
DBLP	No. of hardware blocks/ page for the device
DFIP	Mass storage address
DSIZE	File size in pages
DOPNO	File no. if open

6.3 *INITIALIZATION*

When the NORD-50 monitor is entered, the initialization module is executed to set some important variable to initial values. (Refer to Figure 4.1 and Section 6.2.)

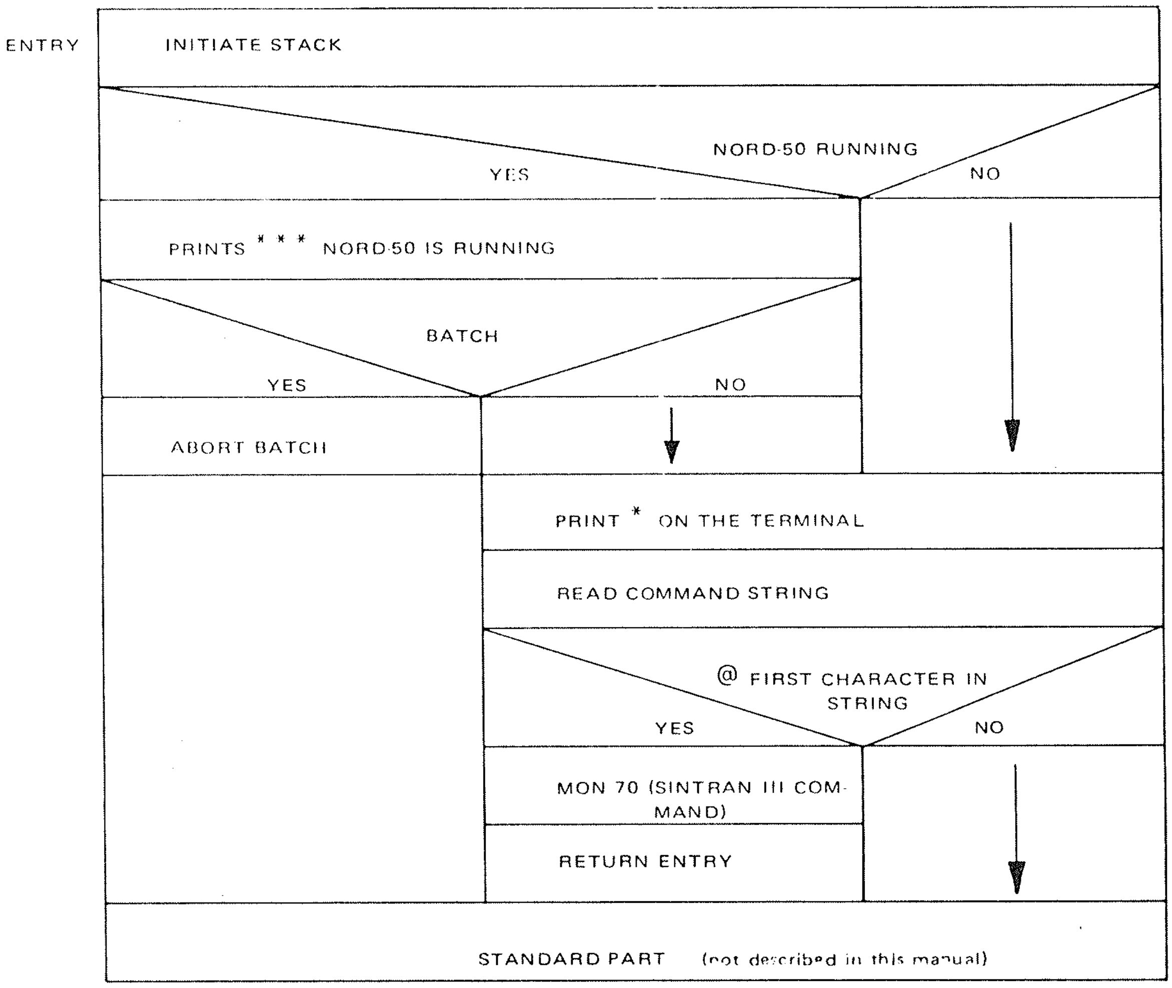

DATF ← A REGISTER (POINTER TO NORD-50 DATAFIELD) , N5LOG ← T REGISTER
INITIATE RFIL, WFILE, UTIME, TCDEV
INITIATE B REGISTER TO POINT TO THE BASEFIELD
INITIATE STACK POINTER
PRINT 'NORD-50 MONITOR' ON THE TERMINAL
CLEAR THE FILE NUMBER CONVERSION TABLE
CLEAR BLOCK 0 BIT MAP
CLEAR DIRECT FILE TABLE
FORMA ← 5
BRKMF, CURSEG, HFLAG AND CPT IS ZEROED
SEGMENT TABLE OK
YES
NO
MEMOK = 1
MEMOK = 0
COLD START FLAG ≠ 0
YES
NO
CALL RELEASE MEMORY
COLD START FLAG ← 0
MEMOK = 1
YES
NO
FIX SEGMENTS USED
BATCH
YES
NO
CALL MASTER CLEAR
COMMAND PROCESSOR ENTRY

6.4 *COMMAND PROCESSOR*

The standard part of the NORD-50 Monitor command processor is quite similar to the one used in the NORD File System.

The command processor entry:

(Refer to Figure 4.1 and Section 6.2.)


ENTRY
INITIATE STACK
NORD-50 RUNNING
YES
NO
PRINTS * * * NORD-50 IS RUNNING
BATCH
YES
NO
ABORT BATCH
PRINT * ON THE TERMINAL
READ COMMAND STRING
@ FIRST CHARACTER IN STRING
YES
NO
MON 70 (SINTRAN III COMMAND)
RETURN ENTRY
STANDARD PART (not described in this manual)

6.5 *NORD-50 MONITOR COMMAND ROUTINES*

When the command processor in the NORD-50 monitor has decided which command was given the control is given to the right command (refer to Figure 4.1).

In the following, the command routines are ordered according to the listing (except the PLACE/LOAD and RUN coommands which are described first).

6.5.1 *The PLACE/LOAD <file name or file number>*

The PLACE/LOAD command moves the executable NORD-50 program file prepared by the NORD-50 loader into the NORD-50 memory.

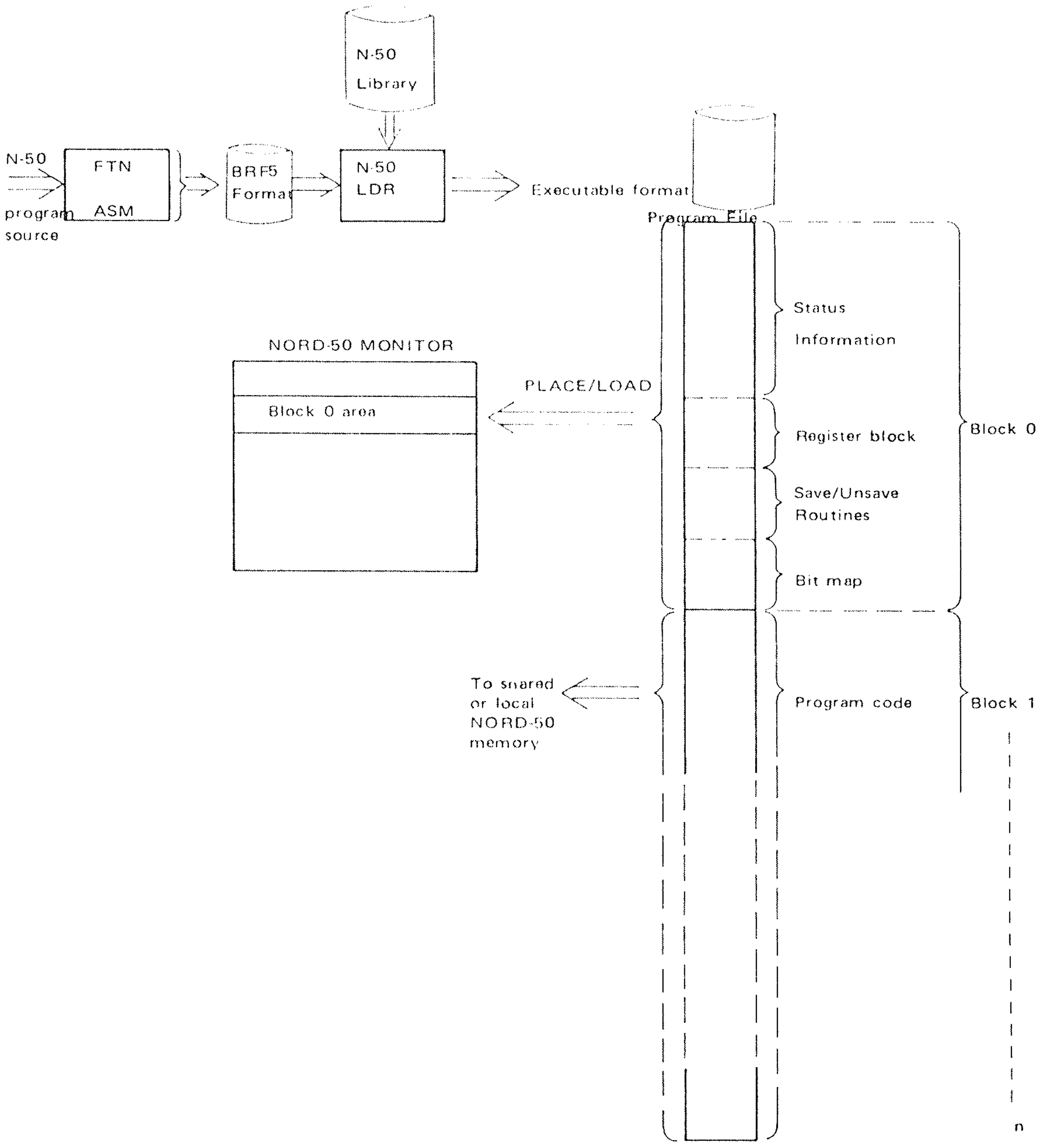
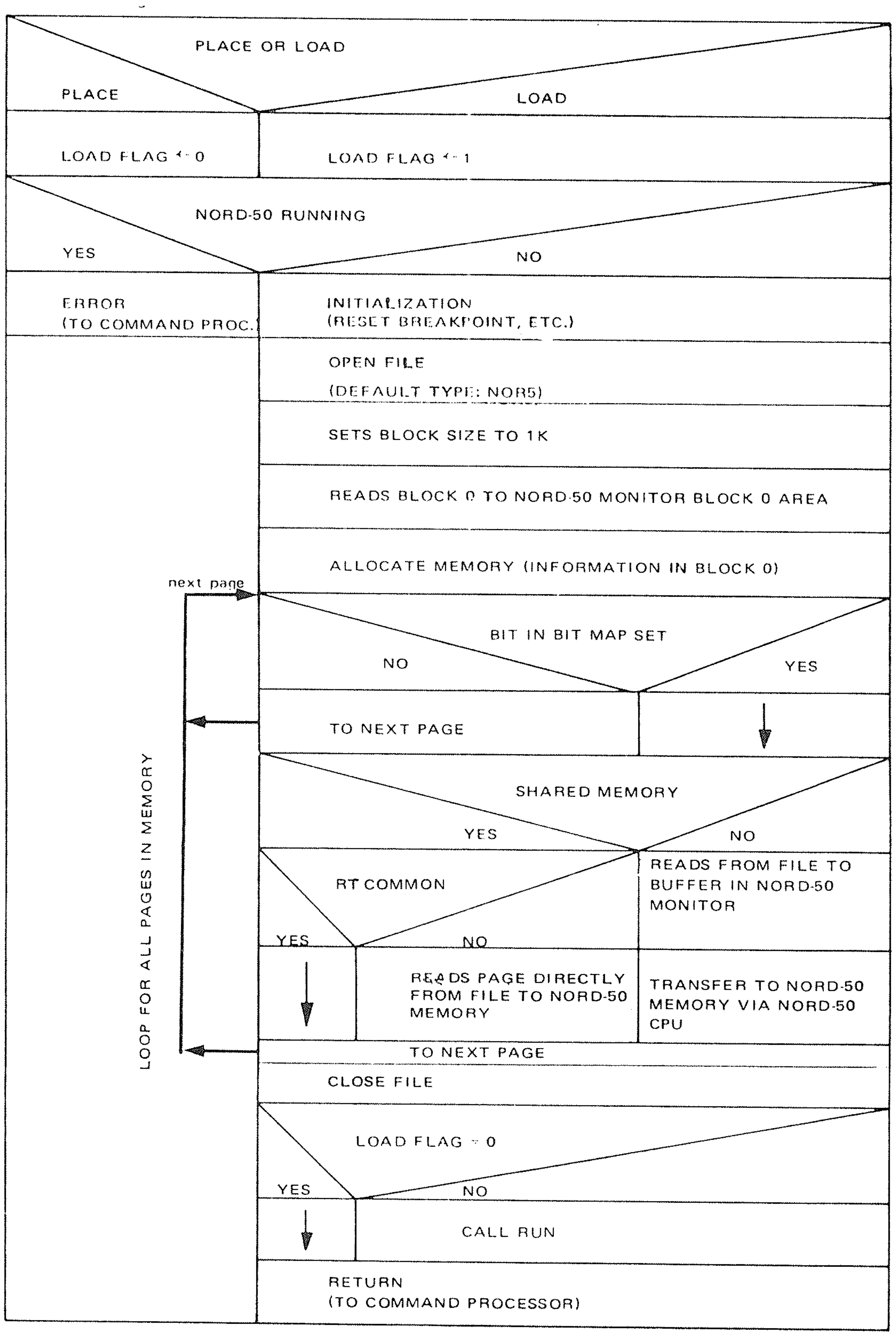


Figure 6.1: The PLACE/LOAD command moves Block 0 from the File to the Data area in the NORD-50 monitor, and allocates memory for the program and moves it to memory. The LOAD command also starts the program by calling the RUN command.

(The program file is generated by the NORD-50 loader and the Block 0 format is the same as described in detail in Sectioon 6.2.2.)

PLACE/LOAD Command:

(Refer to Figure 4.1 and Section 6.2.)



6.5.2 *Run*

The RUN command is used to start or continue execution of a NORD-50 program.

6.5.2.1 SAVE/UNSAVE Routines

Before the NORD-50 program can be started, the actual register contents for the program must be moved from Block 0 area in the NORD-50 Monitor (where it was placed by the PLACE command) to the N-50 CPU, and when the program is terminated (for some reason) the register contents in NORD-50 must be moved from the NORD-50 CPU and saved in Block 0 in the NORD-50 monitor.

This is done by NORD-50 when executing the SAVE/UNSAVE routines which are placed in the NORD-50 Monitor Block 0 area.

Before the SAVE/UNSAVE routines can be executed by NORD-50, the routines must be placed in the NORD-50 memory. This is done by moving the "First Block area" (the first 128 words of the program) in the NORD-50 program to the buffer area in the NORD-50 Monitor (1) and then moving the block 0 in the NORD-50 Monitor to the "First Block area" in the program (2), and vice versa when the routine has been executed. This is performed by the RUN command.

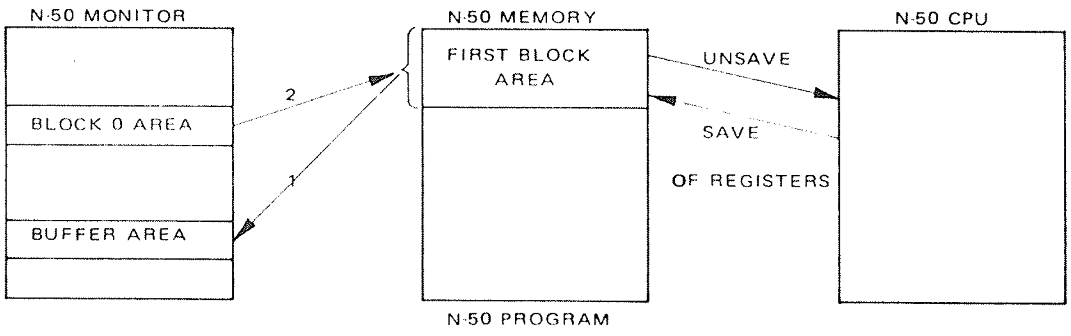


Figure 6.2: Data Transfer for UNSAVE/SAVE of the NORD-50 Register Contents (see Section 1.2)

6.5.2.2 Flow Charts of the RUN Routine

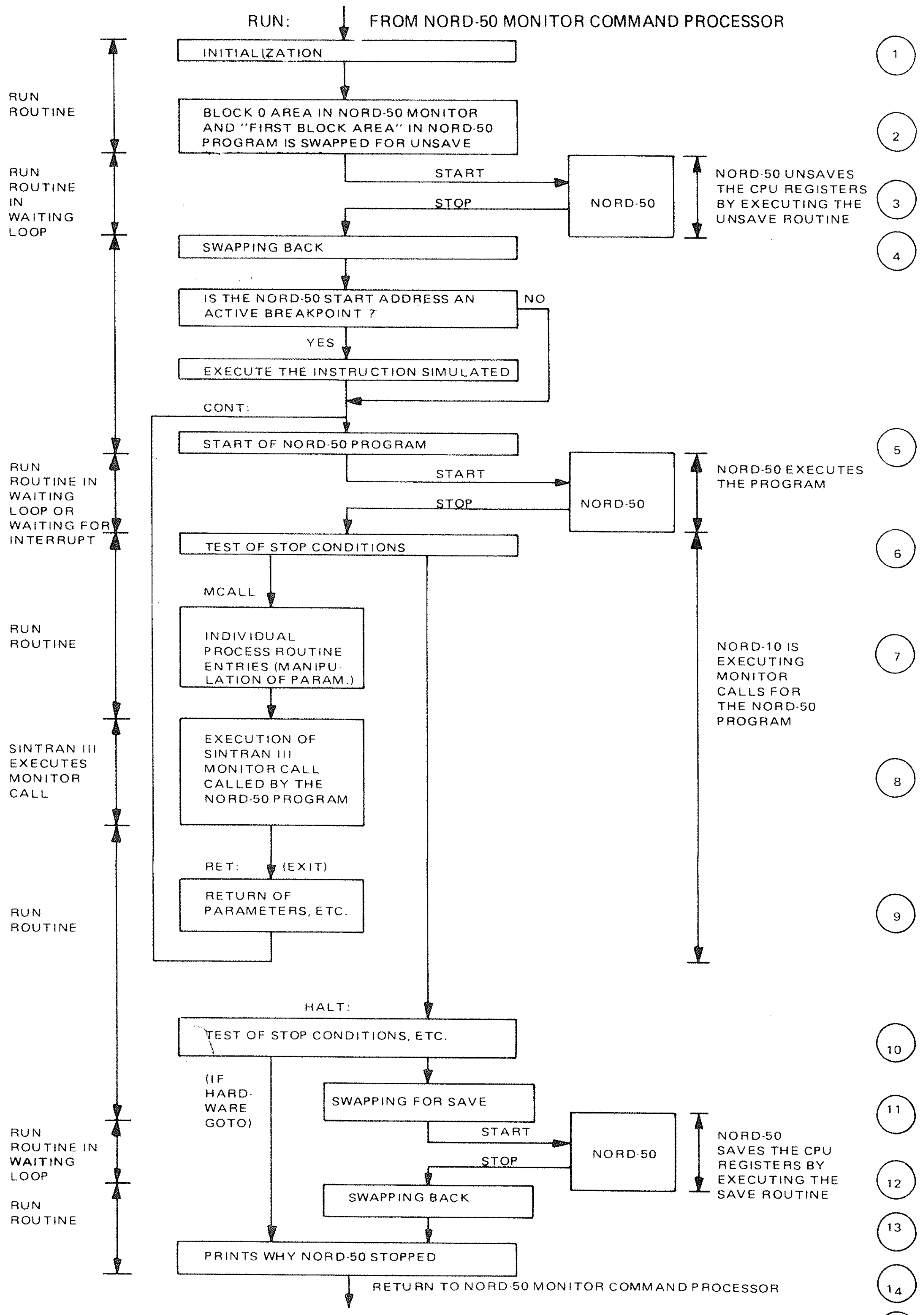

RUN: FROM NORD-50 MONITOR COMMAND PROCESSOR
INITIALIZATION
BLOCK 0 AREA IN NORD-50 MONITOR AND "FIRST BLOCK AREA" IN NORD-50 PROGRAM IS SWAPPED FOR UNSAVE
START
STOP
NORD-50
NORD-50 UNSAVES THE CPU REGISTERS BY EXECUTING THE UNSAVE ROUTINE
SWAPPING BACK
IS THE NORD-50 START ADDRESS AN ACTIVE BREAKPOINT ?
NO
YES
EXECUTE THE INSTRUCTION SIMULATED
CONT:
START OF NORD-50 PROGRAM
START
STOP
NORD-50
NORD-50 EXECUTES THE PROGRAM
TEST OF STOP CONDITIONS
MCALL
INDIVIDUAL PROCESS ROUTINE ENTRIES (MANIPU-LATION OF PARAM.)
EXECUTION OF SINTRAN III MONITOR CALL CALLED BY THE NORD-50 PROGRAM
RET: (EXIT)
RETURN OF PARAMETERS, ETC.
NORD-10 IS EXECUTING MONITOR CALLS FOR THE NORD-50 PROGRAM
HALT:
TEST OF STOP CONDITIONS, ETC.
(IF HARD-WARE GOTO)
SWAPPING FOR SAVE
START
STOP
NORD-50
NORD-50 SAVES THE CPU REGISTERS BY EXECUTING THE SAVE ROUTINE
SWAPPING BACK
PRINTS WHY NORD-50 STOPPED
RETURN TO NORD-50 MONITOR COMMAND PROCESSOR
RUN ROUTINE
RUN ROUTINE IN WAITING LOOP
RUN ROUTINE IN WAITING LOOP OR WAITING FOR INTERRUPT
RUN ROUTINE
SINTRAN III EXECUTES MONITOR CALL
RUN ROUTINE
RUN ROUTINE IN WAITING LOOP
RUN ROUTINE
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Figure 6.3: Main Flow of the RUN Routine

Explanation of the flow chart — Figure 6.3:
(refer to the circled numbers in the figure)

1. When the RUN routine is activated by the command processor in the NORD-50 Monitor (refer to Figure 4.1), some initializations are done. They are described in detail in the flow chart.

2. Then, the first block of the NORD-50 program is transferred to the buffer area in the NORD-50 Monitor for saving, and block in the NORD-50 Monitor data area is transferred to the NORD-50 program area (refer to Figure 6.2).

   As shown in Figure 6.1, during loading of the NORD-50 program the loader places the save/unsave routines in the NORD-50 assembler code in the Block 0 area on the program file. During execution of the PLACE command in NORD-50 Monitor, the Block 0 area on the program file was transferred to the Monitor. This information is now placed in the NORD-50 program area.

3. NORD-50 is then started for execution of the unsave routine. (Refer to Figure 6.2.) The unsave routine transfers the right contents from the register block area to the NORD-50 CPU registers before the NORD-50 program can be executed (see also Figures 5.1 and 6.1 and Section 6.2.2).

4. When NORD-50 has finished execution of the unsave routine, the first block area of the NORD-50 program is transferred back to the NORD-50 Monitor and the buffer area of the NORD-50 Monitor is transferred back to the NORD-50 Monitor is transferred back to the NORD-50 program area.

5. Now the label "CONT" in the RUN routine is reached and NORD-50 starts executing the NORD-50 program.

6. When NORD-50 stops, the RUN routine checks the stop conditions.

7. If NORD-50 stopped because of a Monitor call in the NORD-50 program, the control is given to the right process routine (label "MCALL") where the parameters are transferred to the right NORD-10 registers.

8. Then, the MON instruction is executed and the SINTRAN III Monitor is activated.

9. When the Monitor call is executed, the control is returned to the RUN routine in label "RET" where return parameters are returned to the parameter list, etc.

   Then, the control returns to label "CONT" where NORD-50 is stated to continue execution of the program.

10. If NORD-50 does not stop because of a Monitor call (see point 6), the control is given to label "HALT", where a further check of the stop condition is performed.

11. If NORD-50 does not stop because of a Monitor call, the NORD-50 CPU registers must be saved in the NORD-50 Monitor. The same procedure as described in point 2 is used to transfer the save routine to the NORD-50 memory.

12. NORD-50 is started for execution of the save routine, and the register contents of NORD-50 CPU is transferred to the NORD-50 Memory and placed in the register block area (refer to Figure 6.1 and Section 6.2.2).

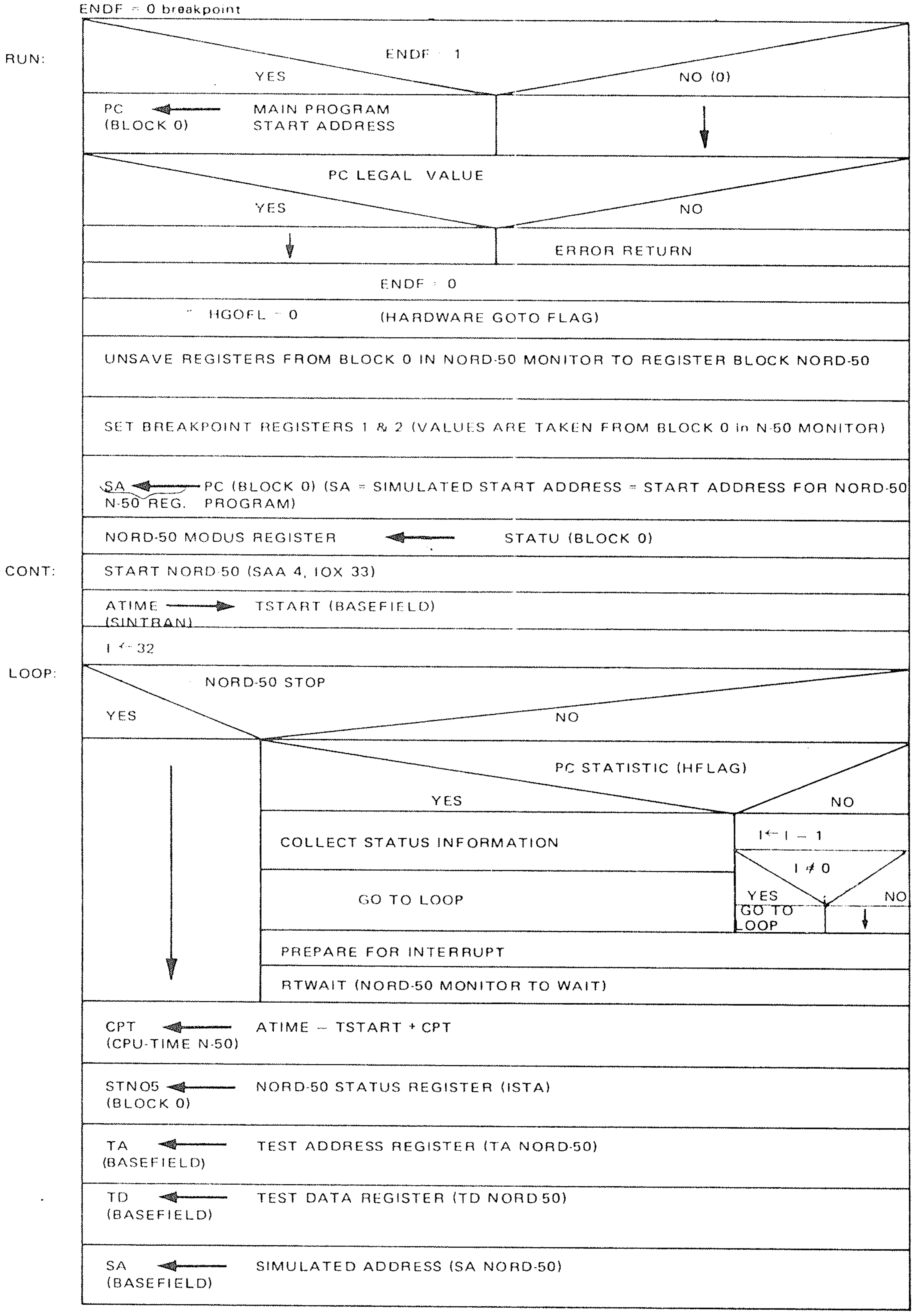
13. When NORD-50 has finished execution of the save routine, the information is transferred from the NORD-50 program area to the NORD-50 Monitor as described in point 4.

14. If the NORD-50 was started by the HARDWARE GOTO command (see Section 3.1), the save/unsave procedures are not performed. (Therefore, the register contents cannot be examined later on.)

15. The RUN routine prints out why NORD-50 stopped, and then returns to the command processor in the NORD-50 Monitor (refer to Figure 4.1) which prints * on the operator terminal, expecting new commands.

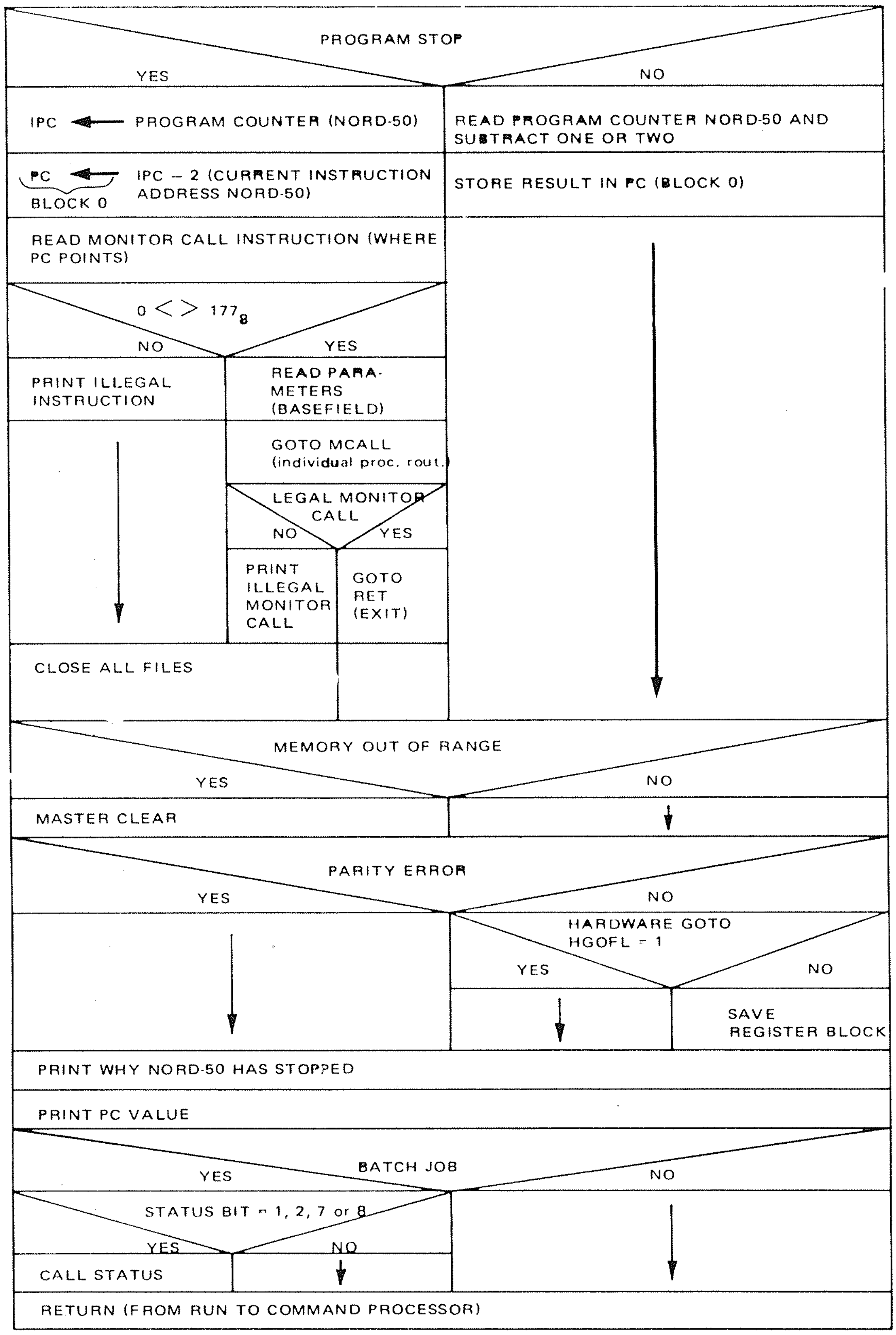
(Refer to Figure 6.3 and Section 6.2.)

RUN ROUTINE:
ENDF = 1 new start
ENDF = 0 breakpoint

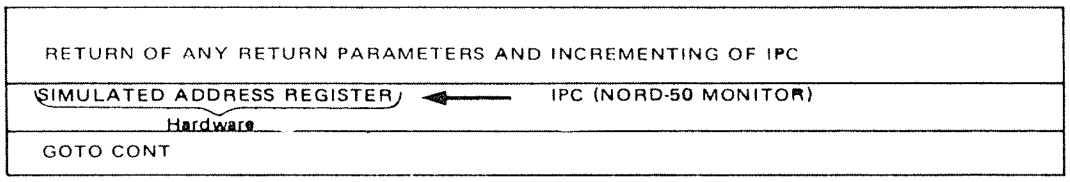


```
RUN:
    ENDF = 1
    YES:
        PC (BLOCK 0) <── MAIN PROGRAM START ADDRESS
    NO (0):
        ↓
    PC LEGAL VALUE
    YES:
        ↓
    NO:
        ERROR RETURN
    ENDF = 0
    HGOFL = 0          (HARDWARE GOTO FLAG)
    UNSAVE REGISTERS FROM BLOCK 0 IN NORD-50 MONITOR TO REGISTER BLOCK NORD-50
    SET BREAKPOINT REGISTERS 1 & 2 (VALUES ARE TAKEN FROM BLOCK 0 in N-50 MONITOR)
    SA (N-50 REG.) <── PC (BLOCK 0) (SA = SIMULATED START ADDRESS = START ADDRESS FOR NORD-50 PROGRAM)
    NORD-50 MODUS REGISTER <── STATU (BLOCK 0)
CONT:
    START NORD-50 (SAA 4, IOX 33)
    ATIME (SINTRAN) ──> TSTART (BASEFIELD)
    I <- 32
LOOP:
    NORD-50 STOP
    YES:
        ↓
    NO:
        PC STATISTIC (HFLAG)
        YES:
            COLLECT STATUS INFORMATION
            GO TO LOOP
        NO:
            I <- I - 1
            I ≠ 0
            YES:
                GO TO LOOP
            NO:
                ↓
        PREPARE FOR INTERRUPT
        RTWAIT (NORD-50 MONITOR TO WAIT)
    CPT (CPU-TIME N-50) <── ATIME - TSTART + CPT
    STN05 (BLOCK 0) <── NORD-50 STATUS REGISTER (ISTA)
    TA (BASEFIELD) <── TEST ADDRESS REGISTER (TA NORD-50)
    TD (BASEFIELD) <── TEST DATA REGISTER (TD NORD-50)
    SA (BASEFIELD) <── SIMULATED ADDRESS (SA NORD-50)
```

HALT:


PROGRAM STOP
YES
NO
IPC ← PROGRAM COUNTER (NORD-50)
READ PROGRAM COUNTER NORD-50 AND SUBTRACT ONE OR TWO
PC BLOCK 0 ← IPC – 2 (CURRENT INSTRUCTION ADDRESS NORD-50)
STORE RESULT IN PC (BLOCK 0)
READ MONITOR CALL INSTRUCTION (WHERE PC POINTS)
0 < > 177₈
NO
YES
PRINT ILLEGAL INSTRUCTION
READ PARAMETERS (BASEFIELD)
GOTO MCALL (individual proc. rout.)
LEGAL MONITOR CALL
NO
YES
PRINT ILLEGAL MONITOR CALL
GOTO RET (EXIT)
CLOSE ALL FILES
MEMORY OUT OF RANGE
YES
NO
MASTER CLEAR
PARITY ERROR
YES
NO
HARDWARE GOTO HGOFL = 1
YES
NO
SAVE REGISTER BLOCK
PRINT WHY NORD-50 HAS STOPPED
PRINT PC VALUE
BATCH JOB
YES
NO
STATUS BIT = 1, 2, 7 or 8
YES
NO
CALL STATUS
RETURN (FROM RUN TO COMMAND PROCESSOR)

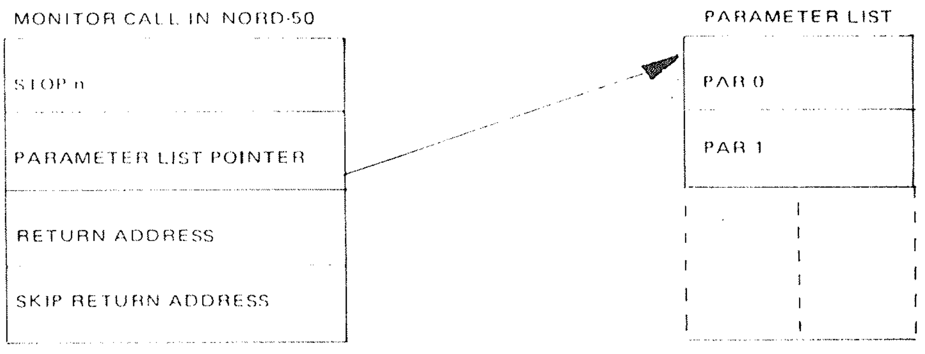
RET:


RETURN OF ANY RETURN PARAMETERS AND INCREMENTING OF IPC
SIMULATED ADDRESS REGISTER ← IPC (NORD-50 MONITOR)
Hardware
GOTO CONT

6.5.2.3 Individual Process Routine Entries (SINTRAN III Monitor Calls)

When NORD-50 programs require the execution of a SINTRAN III monitor call, the monitor call parameters from the NORD-50 program must be transferred to the relevant NORD-10 registers before the Monitor instruction is executed.

Monitor call format in the NORD-50 program:



n = MONITOR CALL NUMBER

For each SINTRAN III Monitor call available to NORD-50 programs, the NORD-50 Monitor contains a process routine to transfer the parameters to/from the NORD-10 registers (refer to Figure 6.3).

SINTRAN III Monitor calls implemented for NORD-50 programs and the relationship between the parameters and the NORD-10 registers:

MON 0 (RTEXT)

Prints STOP 0 and then goes to HALT (does not perform monitor call).

MON 134 (RTEXT)

Sets ENDF = 1 and prints ***END***
Closes all files and goes to HALT.

MON 1 (INBYTE <file number, byte>)

Registers:

T = PAR0
A = PAR1

MON 2 (OUTBYTE <file number>, <byte>)

Registers:

T = PAR0
A = PAR1

MON 3 (SET ECHO MODE <term no.>, <mode>)

Registers:

T = PAR0
A = PAR1

MON 4 (SET BREAK MODE <term no.>, <mode>)

Registers:

T = PAR0
A = PAR1

MON 11 (READ INTERNAL TIME <time>)

Registers:

A = PAR0
D = PAR0

MON 13 (CLEAR INPUT BUFFER <file no.>)

Registers:

T = PAR0
A = PAR1 (error no.)

MON 14 (CLEAR OUTPUT BUFFER <file no.>)

Registers:

T = PAR0
A = PAR1 (error)

MON 43 (CLOSE FILE <file no.>)

Registers:

T = PAR0
A = PAR1 (error)

MON 45 (TYPRING <file no./error message>, <typring>, <status bits>, <actual file no.>)

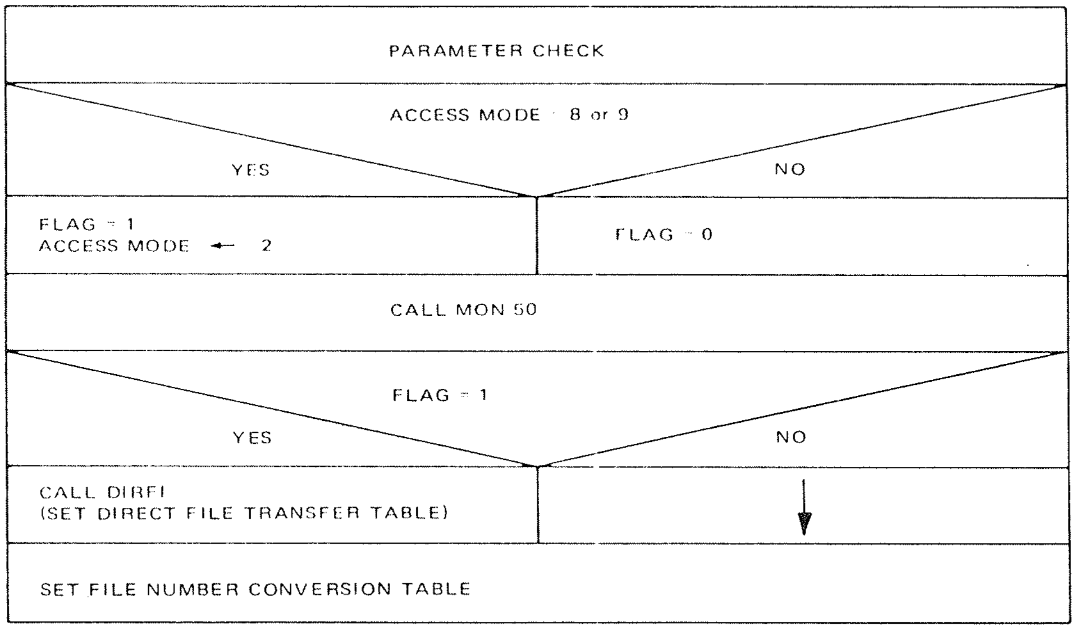
PAR0 = error message if error

Return some information of an opened file in the third last parameter (special NORD-50 monitor call).

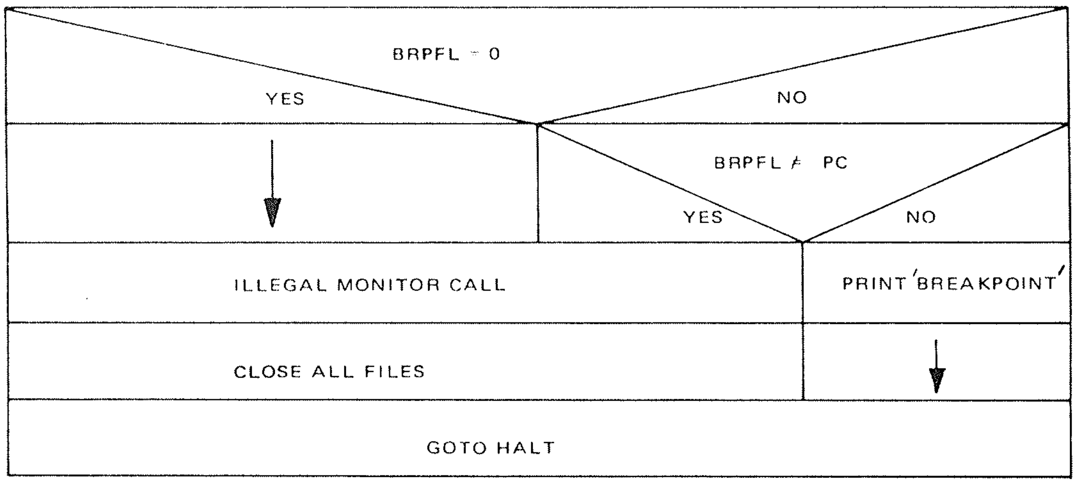
MON 50 (OPEN FILE <file name>, <access mode>, <default type>, <conn. file number or 0>)

Registers:

X = PAR0
T = PAR1
A = PAR2



MON 51 (BREAKPOINT)



MON 62 (READ MAX. BYTE POINTER <file no>, <pointers>)

Registers:

T = PAR0

A and/or AD = PAR1
 if error: A = error code
 if OK: AD = number of bytes

MON 64 (WRITE SYSTEM ERROR MESSAGE <number>)

Register:

A = PAR0

MON 65 (64 AND STOP <number>)

Register:

A = PAR0

MON 66 (GET INPUT BUFFER SIZE <file no.>, <no. of characters>)

Registers:

T = PAR0
A = PAR1

MON 67 (OUTPUT BUFFER SIZE <file no.>, <no. of characters>)

Registers:

T = PAR0
A = PAR1

MON 70 (COMND <string>)

PAR0 — N = string

MON 73 (SET MAXIMUM BYTE COUNTER <file no.>, <byte pointer>)

Registers:

Same as for MON 62.

MON 74 (SET BYTE COUNTER <file no.>, <byte pointer>)

Registers:

Same as for MON 62.

MON 75 (READ BYTE POINTER <file no.>, <byte pointer>

Registers:

T = PAR0

AD = PAR1
if error: A = error code
if OK: AD = byte pointer

MON 76 (SET BLOCK SIZE <file no.>, <block size>)

Registers:

Same as for MON 62.

Block size in NORD-50 words.

MON 77 (SET BLOCK POINTER <file no.>, <block pointer>)

Registers:

Same as for MON 62.

Standard FORTRAN Monitor Calls:

MON 100, 101, 102, 103, 104, 105, 106, 107, 110, 111, 112, 122, 123, 124, 125, 126, 127, 130, 136, 137, 140, 141.

Register A →	PAR 0 (eventual function return)	
	PAR 1	
	PAR 2	Parameters
	.	
	.	

MON 113 (CLOCK <basic>, <second>, <minute>, <hour>, <day>, <month>, <year>)

PAR0	basic time unit
PAR1	seconds
PAR2	minutes
PAR3	hour
PAR4	day
PAR5	month
PAR6	year

MON 114 (TIME USED <N-50 CPU time used since N-50 Monitor entered>)

Register:

AD = PAR0

MON 117 (READ FILE <file no.>, <return flag>, <address>, <block no.>, <words>)

Register A →	PAR0	
	PAR1	file number
	PAR2	return flag
	.	
	.	

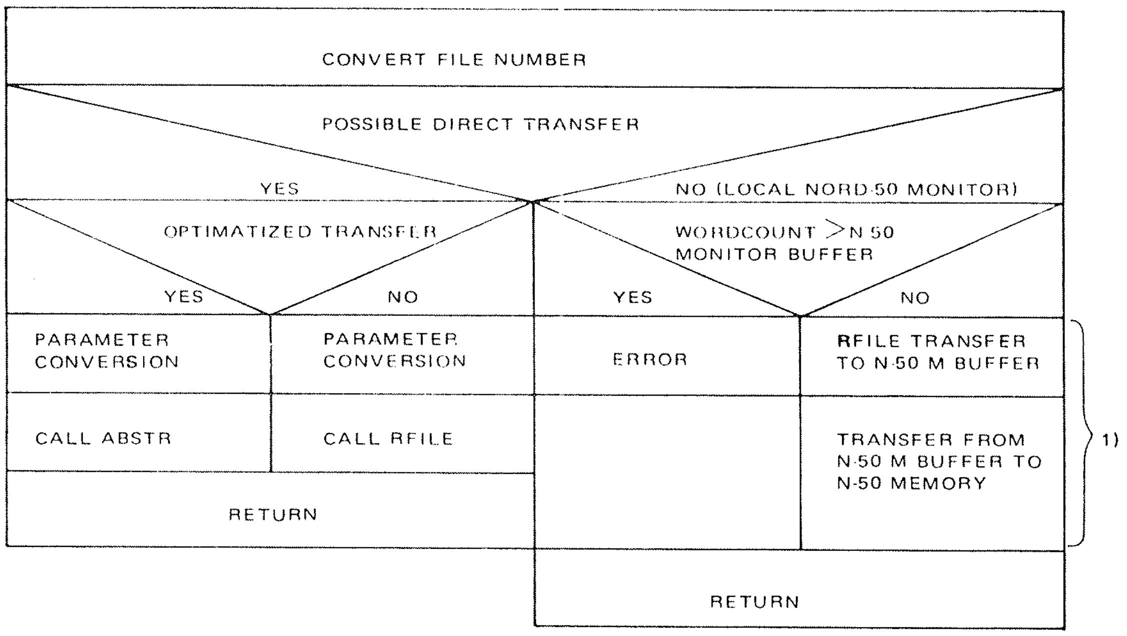
Word count in NORD-50 words.

MON 120 (WRITE FILE <file no.>, <return flag>, <address>, <block no.>, <words>)

Registers:

Same as for MON 117.

MON 117, 120, 121:



1) opposite transfer if WFILE (MON 120)

MON 131 (ABSTR <unit>, <function>, <address>, <block>, <no. of words>)

PAR0	error message
PAR1	unit
PAR2	function

MON 135 (RT WAIT)

MON 143 (RSIO <input file>, <mode>, <output file>)

Registers:

T = PAR0
A = PAR1
D = PAR2

MON 144 (MAGTP <function>, <memory address>, <logical no.>, <max. words>, <words read>)

PAR0	error message
PAR1	function

Maximum words and words read in NORD-50 words.

MON 145 (ACM <logical unit>, <function>, <memory address>, <DMA address> <word count>)

PAR0	error message
PAR1	function

MON 161 (INSTR <logical no.>, <memory address>, <max. words>, <terminator>)

PAR0	function

MON 162 (OUTST <logical no.>, <memory address>, <number>)

PAR 0	function

6.5.2.4 *The Other Command Routines*

The remaining NORD-50 Monitor commands described in Chapter 3 are very simple to understand. For further details refer to the listing of the Monitor and the data part definition — Section 6.2 in this manual.

In the listing the commands are ordered as shown below (and for some of them a flow chart is shown).

CPU TIME (prints the variable CPT in the base field)

ZERO-MEMORY (see Breakpoint registers in block 0)

HISTOGRAM-WRITE (prints the contents of the Histogram parameters in the base field)

HISTOGRAM-ON (see base field)

HISTOGRAM-OFF (see base field)

SET-MEMORY (see segment table, Section 6.2.3)

CC

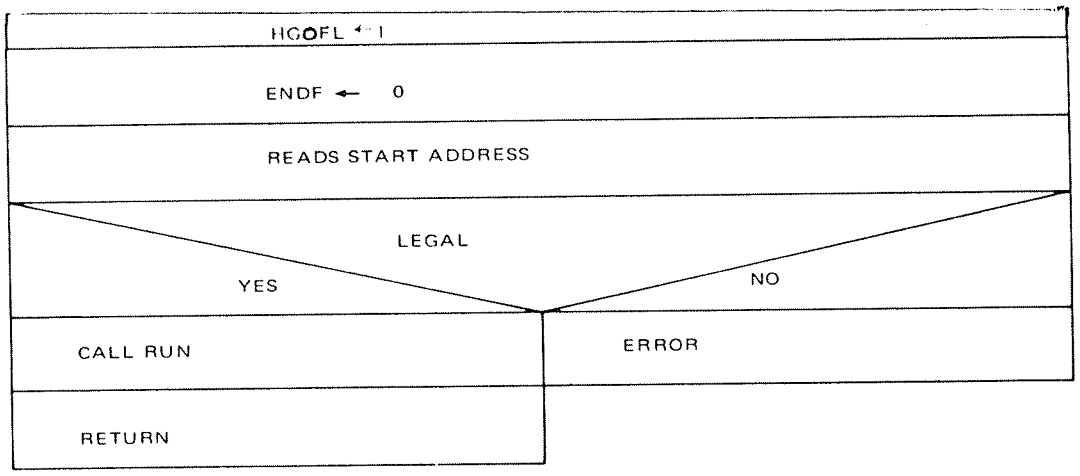
LIST-MEMORY (prints the contents of segment table!

RESERVE-MEMORY (see segment table, Section 6.2.3)

RELEASE-MEMORY (see segment table, Section 6.2.3)

HARDWARE-GOTO

This command starts execution of a NORD-50 program in the specified address. The NORD-50 register block is not saved when this command is used (refer to Section 6.5.2) and therefore the register contents cannot be examined (see the symbol HGOFL in the base field). This command is usually used when debugging NORD-50 hardware.



CARRY (prints the block 0 variable "carry" in which the contents of NORD-50 "carry flip-flop" is saved)

REMAINDER (the contents of the NORD-50 remainder register, which is saved in the block 0 area in variable EXT1, is printed)

OVERFLOW (the contents of the NORD-50 overflow register, which is saved in the block 0 area in the variable EXT2, is printed)

FORMAT (see the program variable area in the base field, symbol FORMA)

PRINT (prints the contents of the NORD-50 Memory addresses or register interval. The NORD-50 registers are saved in the Block 0 area, symbol REGAR.)

C (single instruction execution, see the program variable in the base field)

HELP (prints the contents of the NORD-50 Monitor command table)

BREAK-CONDITIONS (see the block 0 area – BP, BQ – and the segment table)

STATUS (see the block 0 area and the program variable are in the base field)

LOOK-AT (examine and deposit of the NORD-50 Memory address or register, see the block 0 area and "FORMA" in the program variable area)

SINTRAN or EXIT (see also the segment table and the program variable area)

HARDWARE-STATUS (reads and prints information from NORD-50 hardware communication module registers – STATUS, PC, SA, TA, TD – see Figure 1.2)

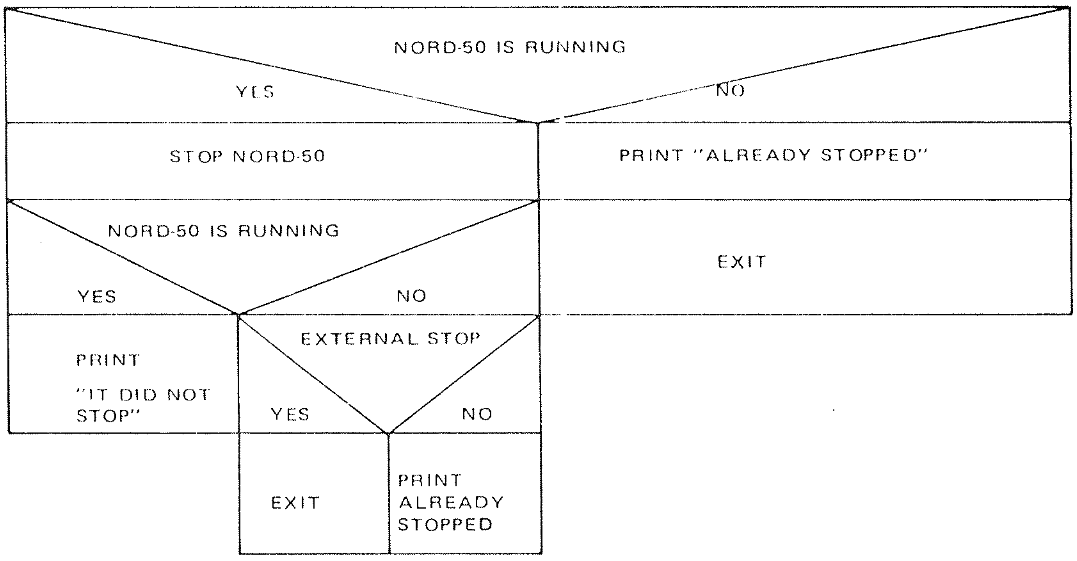
TEST-MEMORY

LOOP-ON

LOOP-OFF

MASTER-CLEAR

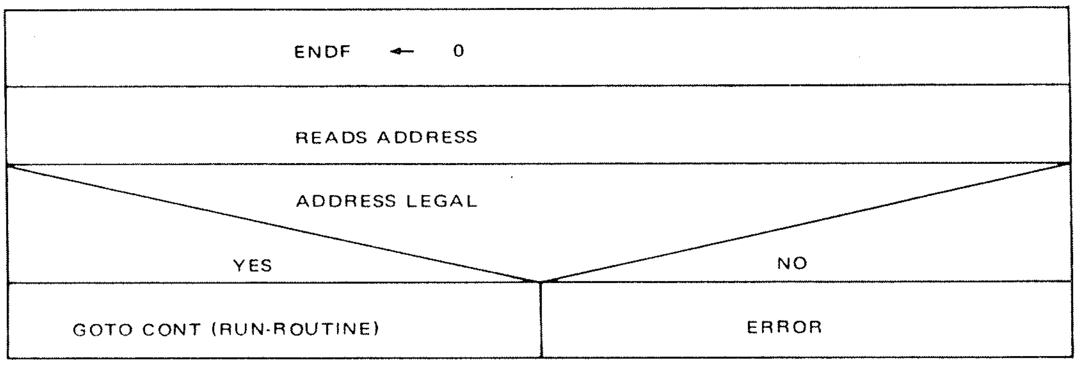
STOP (stops the NORD-50 if it is running)



SAVE (see the block 0 area)

OPEN-FILE (for access modes D and DC, see also the direct file table in Section 6.2.4)

GOTO (see flow chart below and the program variable area)



APPENDIX A

NORD-50 MONITOR COMMAND SUMMARY

COMMAND:	*PARAMETERS:*	*SHORT DESCRIPTION:*	*SECTION:*
BREAK-ADDRESS	Address	Sets or moves the breakpoint to the NORD-50 address	3.1.5.2
BREAK-CONDITIONS	BP address, BQ address and break conditions	Sets the NORD-50 breakpoint registers and break conditions	
BREAK-LINE	Line no. [program unit (optional)]	Sets or moves the breakpoint to the line	3.1.5.1
BREAK-NUMBER	Number	Sets breakpoint number for a breakpoint	3.1.5.10
C		Single instruction execution	3.1.5.3
CARRY		Prints the NORD-50 carry flip flop	3.1.7.4
CC	Comment line	Comment	
CHANGE	Variable, program unit, value	Changes the value of the variable	3.1.5.23
CLEAR-BREAKS		Restores all breakpoints, exhibit points and trap	3.1.5.12
CLOSE-FILE	File number	Closes a file	3.1.4.2
CONDITIONAL-BREAK-ADDRESS	Address, low limit, high limit	Conditional break in an octal address	3.1.5.5
CONDITIONAL-BREAKLINE	Line no., program unit name, variable name, program unit name, low limit, [high limit]	Conditional break in a line	3.1.5.4
CPU-TIME		Prints NORD-50 CPU time used	3.1.6.4
DISPLAY	Variable name, [program unit]	Lists the address and value	3.1.5.28
DUMP-ADDRESS	Low address, high address, [output file]	The contents of addresses or registers is printed	3.1.5.28
DUMP-ALL-VARIABLES	[output file]	Prints all variables in all units	3.1.5.27
DUMP-ARRAY	Array name, program name, [output file]	Prints values of all the array elements	3.1.5.25
DUMP-VARIABLE	Program unit name, [output file]	Prints all the variables in the program unit	3.1.5.26
ENTRIES-DEFINED	[output file]	Prints all symbols defined	3.1.5.20
ENTRIES-UNDEFINED	[output file]	Prints all symbols undefined	3.1.5.21
EXHIBIT-ADDRESS	Address in program, address of variable	Sets or moves the exhibit point to the NORD-50 address	3.1.5.7
EXHIBIT-CLOSE		Closes the exhibit file	3.1.5.9
EXHIBIT-LINE	Line no., program unit, variable, [program unit]	Sets or moves the exhibit point to the line	3.1.5.6
EXHIBIT-OPEN	output file name	Opens a file for output from exhibit points	3.1.5.8
EXIT		Returns to SINTRAN. Releases memory and NORD-50.	3.1.8.6
FORMAT	Formats	Sets output formats to be used	3.1.5.29
GOTO	Address	Starts execution of NORD-50 program in address	3.1.3.4
HARDWARE-GOTO	Address	Starts execution in address. Registers are nore saved	3.1.7.1
HARDWARE-STATUS		Prints some information about NORD-50 status	3.1.7.3
HELP	Command, [output file]	Prints the commands available	3.1.8.9
HISTOGRAM-OFF		Stops program counter sampling	3.1.6.2
HISTOGRAM-ON	First address, words/-channel, no. of channels	Clears Histogram table and starts sampling PC	3.1.6.1
HISTOGRAM-WRITE	[output file]	Prints the Histogram	3.1.6.3
LIST-BREAKS		List information about the active breakpoints	3.1.5.13

COMMAND:	*PARAMETERS:*	*SHORT DESCRIPTION:*	*SECTION:*
LIST-MEMORY	[output file]	Prints the memory configuration specfiied	3.1.8.2
LIST-TITLE		The NORD-50 monitor's title is printed	3.1.6.9
LIST-UNIT	[output file]	Prints the names and additional information of the program	3.1.5.19
LOAD-AND-GO	File name or number	Loads and starts the NORD-50 program	3.1.3.2
LOOK-AT	Address or register	Examine and change addresses or registers	3.1.5.24
LOOP-OFF		Turns off the examineddeposit loop for TEST-MEM	3.1.7.9
LOOP-ON		Turns on the examineddeposit loop for TEST-MEM	3.1.7.8
MASTER-CLEAR		Brings the NORD-50 out of hang-up	3.1.7.2
MEMORY-MAP	[output file]	Prints a memory map of the NORD-50 system	3.1.5.10
NEST		Prints the current dynamic subroutine call nesting	3.1.5.18
OPEN-FILE	Name, number, access	Opens and connects a file number	3.1.4.1
OVERFLOW		Prints the NORD-50 overflow register	3.1.7.6
PLACE	file name or number	Loads the NORD-50 program	3.1.3.1
PRINT	Low address, high address, [output file]	The contents of address or registers is printed	
QUIT		Returns to SINTRAN. Memory reservation and protect setting is not changed	3.1.8.7
RELEASE-MEMORY		Both static and dynamic memory is released	3.1.8.4
REMAINDER		Prints the NORD-50 remainder register	3.1.7.5
RESERVE-MEMORY		Static memory is reserved for NORD-50 use	3.1.8.3
RESET-BREAK		Restores a breakpoint or an exhibit point	3.1.5.11
RESET-TRAP		Restores a trap	3.1.5.17
RUN		Starts execution of a NORD-50 program	3.1.3.3
SAVE	File name or number	Saves the NORD-50 memory and some additional information	3.1.5.32
SET-MEMORY	NORD-10 page, segment no., size, type	Specifies the NORD-50 memory	3.1.8.1
SINTRAN		Same as EXIT	3.1.8.6
STATUS		Prints some information about the NORD-50 status	3.1.5.31
STOP		Stops the NORD-50 if it is running	3.1.8.5
TEST-MEMORY		Tests the NORD-50 memory	3.1.7.7
TRAP-ADDRESS	Low address, high address, conditions	Sets a trap on access to a memory area	3.1.5.16
TRAP-LINE	Low line no., high line no., program unit, conditions	Sets a trap in executable statements	3.1.5.14
TRAP-VARIABLE	Variable program unit, conditions	Sets a trap on access to the variable	3.1.5.15
X-FORMAT	Formats	Sets output format to be used by some special commands	3.1.5.30
ZERO-MEMORY		Stores zero in the NORD-50 memory	3.1.5.34

APPENDIX B

MONITOR CALLS IMPLEMENTED IN NORD-50 MONITOR

Monitor Call:	*Command:*	*Parameters:*	*Section:*
0	RTEXT	No parameters	6.5.2.3
1	INBT	File number, byte	6.5.2.3
2	OUTBT	File number, byte	6.5.2.3
3	ECHOM	Bit map, mode	6.5.2.3
4	BRKM	Bit map, mode	6.5.2.3
5 - 7	Not Implemented in NORD-50		
10	Not Implemented in NORD-50		
11	TIME	time	6.5.2.3
12	NOT USED		
13	CIBUF	File number	6.5.2.3
14	COBUF	File number	6.5.2.3
15 - 42	NOT USED		
43	CLOSE	File number	6.5.2.3
44	NOT USED		
45	TYPRING	File no., type ring, status, actual file no.	6.5.2.3
46 - 47	Not Implemented in NORD-50		
50	OPEN	File name, access mode, default type, conn. file no. of 0	6.5.2.3
51	BREAKPOINT	No parameters	6.5.2.3
52 - 61	NOT USED		
62	RMAX	File no., pointers	6.5.2.3
63	NOT USED		
64	ERMSG	number	6.5.2.3
65	QERMS	number	6.5.2.3
66	ISIZE	file no., no. of char.	6.5.2.3
67	OSIZE	file no., no. of char.	6.5.2.3
70	CMND	PAR0-N = string	6.5.2.3
71 - 72	NOT USED		
73	SMAX	file no., byte pointer	6.5.2.3
74	SETBY	file no., byte pointer	6.5.2.3
75	REABT	file no., byte pointer	6.5.2.3
76	SBIZ	file no., block size	6.5.2.3
77	SETBC	file no., block pointer	6.5.2.3
100	RT	Standard FORTRAN	
101	SET	Standard FORTRAN	
102	ABSET	Standard FORTRAN	
103	INTV	Standard FORTRAN	
104	HOLD	Standard FORTRAN	
105	ABORT	Standard FORTRAN	
106	CONCT	Standard FORTRAN	
107	DSCNT	Standard FORTRAN	
110	PRIOR	Standard FORTRAN	
111	UPDAT	Standard FORTRAN	
112	CLADJ	Standard FORTRAN	
113	CLOCK	Integer array (7 words)	6.5.2.3
114	TUSED	CPU time used	6.5.2.3

Monitor Call:	*Command:*	*Parameters:*	*Section:*
115 - 116	Not Implemented in NORD-50		
117	RFILE	File no., return flag, address, block no., words	6.5.2.3
120	WFILE	File no., return flag, address, block no., words	6.5.2.3
121	WAITF		
122	RESRV	Standard FORTRAN	
123	RELES	Standard FORTRAN	
124	PRSRV	Standard FORTRAN	
125	PRLS	Standard FORTRAN	
126	DSET	Standard FORTRAN	
127	DABST	Standard FORTRAN	
130	DINTV	Standard FORTRAN	
131	ABSTR	unit, function, address, block, no. of words	6.5.2.3
132 - 133	Not Implemented in NORD-50		
134	RTEXT	No parameter	6.5.2.3
135	RTWT		
136	RTON	Standard FORTRAN	
137	RTOFF	Standard FORTRAN	
140	WHDEV	Standard FORTRAN	
141	IOSET	Standard FORTRAN	
142	Not Implemented in NORD-50		
143	RSIO	Input file, mode, output file	6.5.2.3
144	MAGTP	Func., mem. adr., log. no., max. word, word read	
145	ACM	Log, unit, func. mem. adr., DMA adr., word count	
146 - 160	Not Implemented in NORD-50		
161	INSTR	Log. no., mem. adr., max. no., terminator	
162	OUTST	Log. no., mem. adr., no.	
163 - 166	Not Implemented in NORD-50		
167	NOT USED		
170 - 177	User Monitor Calls US0-US7 may be used if standard FORTRAN parameter list in NORD-10		

NORSK DATA A.S.
Lørenveien 57 - Postboks 163, Økern
OSLO 1

COMMENT AND EVALUATION SHEET

NORD-50 MONITOR – User's Guide and System Documentation
NOVEMBER 1978 Publ. No. ND-60.076.02

In order for this manual to develop to the point where it best suits your needs, we must have your comments, corrections, suggestions for additions, etc. Please write down your comments on this pre-addressed form and post it. Please be specific wherever possible.

FROM ______________________________

____

____

– we make bits for the future

NORSK DATA A.S LØRENVEIEN 57 OSLO 5 NORWAY PHONE: 21 73 71 TELEX: 18284