


Norsk Data



SINTRAN III **Timesharing/Batch Guide**

ND-60.132.03



SINTRAN III

Timesharing/Batch Guide

ND-60.132.03

NOTICE

The information in this document is subject to change without notice. Norsk Data A.S. assumes no responsibility for any errors that may appear in this document. Norsk Data A.S. assumes no responsibility for the use or reliability of its software on equipment that is not furnished or supported by Norsk Data A.S.

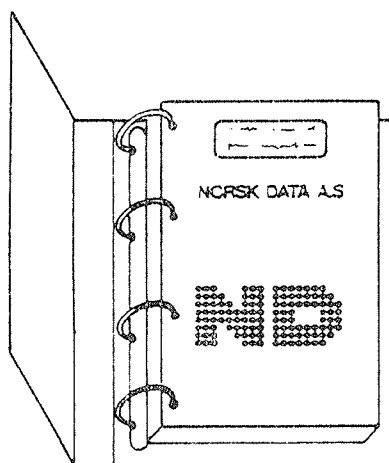
The information described in this document is protected by copyright. It may not be photocopied, reproduced or translated without the prior consent of Norsk Data A.S.

Copyright © 1982 by Norsk Data A.S.

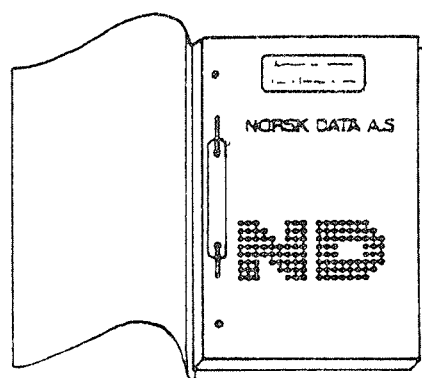
This manual is in loose leaf form for ease of updating. Old pages may be removed and new pages easily inserted if the manual is revised.

The loose leaf form also allows you to place the manual in a ring binder (A) for greater protection and convenience of use. Ring binders with 4 rings corresponding to the holes in the manual may be ordered in two widths, 30 mm and 40 mm. Use the order form below.

The manual may also be placed in a plastic cover (B). This cover is more suitable for manuals of less than 100 pages than for large manuals. Plastic covers may also be ordered below.



A Ring Binder



B Plastic Cover

Please send your order to the local ND office or (in Norway) to:

Norsk Data A.S
Graphic Center
P.O. Box 25, Bogerud
0621 Oslo 6, Norway

ORDER FORM

I would like to order

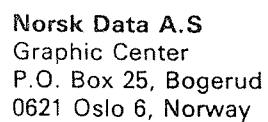
..... Ring Binders, 30 mm, at nkr 20,- per binder

..... Ring Binders, 40 mm, at nkr 25,- per binder

..... Plastic Covers at nkr 10,- per cover

Name
Company
Address
.....
City

Sintran III Timesharing/Batch Guide
Publ.No. ND-60.132.03



Manuals can be updated in two ways, new versions and revisions. New versions consist of a complete new manual which replaces the old manual. New versions incorporate all revisions since the previous version. Revisions consist of one or more single pages to be merged into the manual by the user, each revised page being listed on the new printing record sent out with the revision. The old printing record should be replaced by the new one.

New versions and revisions are announced in the Customer Support Information (CSI) and can be ordered as described below.

The reader's comments form at the back of this manual can be used both to report errors in the manual and to give an evaluation of the manual. Both detailed and general comments are welcome.

These forms and comments should be sent to:

Documentation Department
Norsk Data A.S
P.O. Box 25, Bogerud
0621 Oslo 6, Norway

Requests for documentation should be sent to the local ND office or (in Norway) to:

Graphic Center
Norsk Data A.S
P.O. Box 25, Bogerud
0621 Oslo 6, Norway

Preface:

The Product

This manual documents the H version of SINTRAN III VS, VSE and VSE-500.

The Reader

This manual is intended for the user who needs an explanation of the time-sharing and batch facilities in SINTRAN III.

Prerequisite Knowledge

The reader should be familiar with the contents of the manual

SINTRAN III INTRODUCTION (ND-60.125)

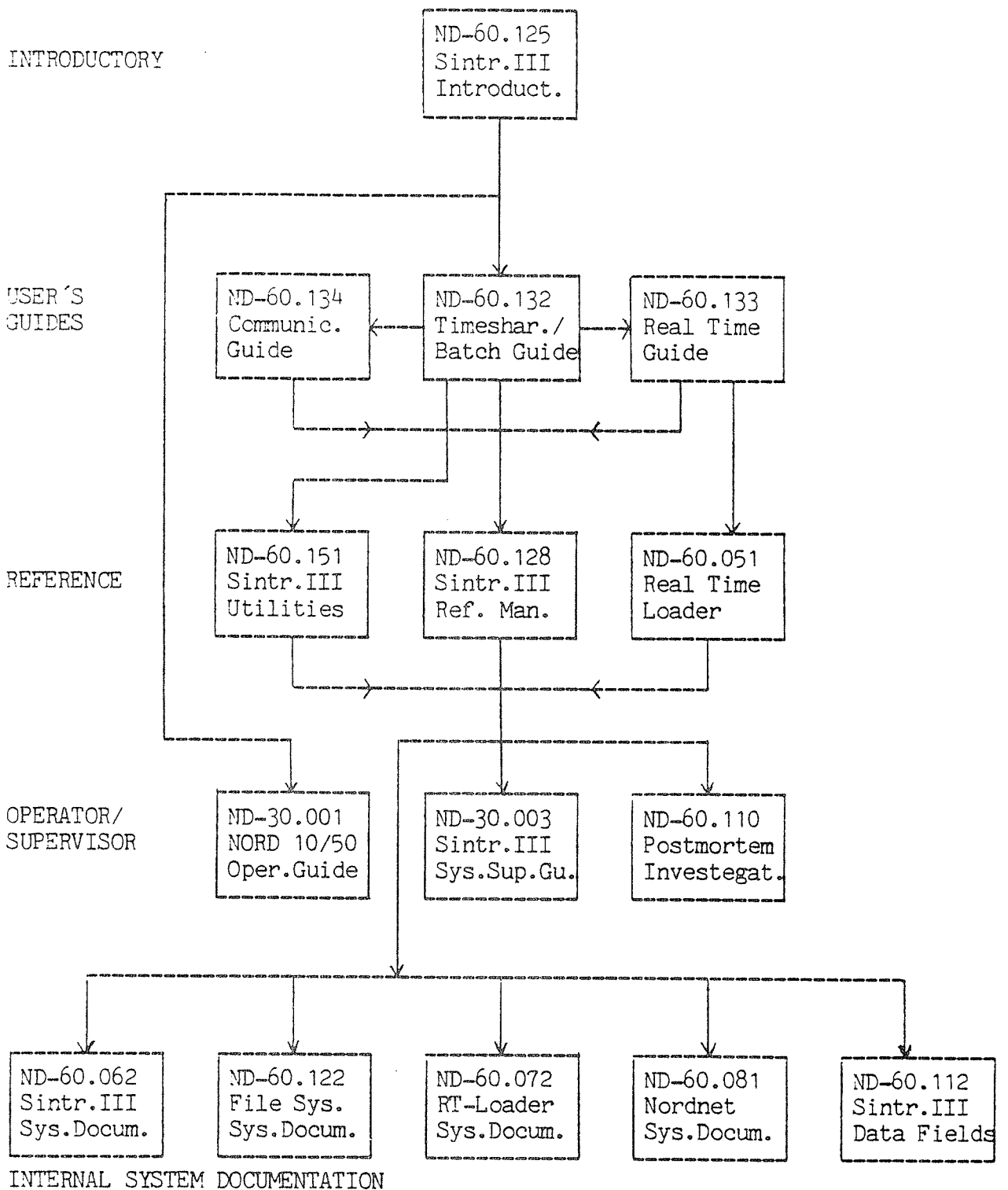
The Manual

This manual shows command and monitor calls available for the time-sharing and batch user. The commands and monitor calls are ordered according to functional category as opposed to the SINTRAN III REFERENCE MANUAL where they are fully documented in alphabetical order.

Related manuals guiding the users are:

- SINTRAN III REAL TIME GUIDE (ND-60.133)
- SINTRAN III COMMUNICATION GUIDE (ND-60.134)
- SINTRAN III REFERENCE MANUAL (ND-60.128)
- SINTRAN III SYSTEM SUPERVISOR (ND-60.103)
- SINTRAN III UTILITIES (ND-60.151)
- SINTRAN III RT LOADER (ND-60.051)

vii
SINTRAN III/VS



SINTRAN III/RT

ND-60.082
Sin.III/RT
Ref. Man.

TABLE OF CONTENTS

Section	Page
1. Introduction	1
1.1. General	1
1.2. User Categories	1
1.2.1. Time-Sharing and Batch Users	1
1.2.2. The Real-Time User	2
1.2.3. System Supervisor	2
1.3. Commands and Monitor Calls	2
1.4. Hardware Environment	3
1.5. SINTRAN III Subsystems	4
1.6. Documentation	4
2. Using The Terminal	5
2.1. The ND Terminals	5
2.2. Logging In	6
2.3. Executing Commands and Programs	8
2.4. Writing SINTRAN III Commands	9
2.4.1. Single Line Commands	9
2.4.2. Abbreviation in Commands	10
2.4.3. Use of Asterisk (*)	10
2.4.4. Default Values	10
2.4.5. Prompting Line Commands	11
2.4.6. Correcting Typing Errors	11
6.1. Correcting the Current Line	11
6.2. Editing the Previous Line	12
2.5. Error Messages from SINTRAN III	13
2.6. Listing Information about the System	14
2.7. @LOGOUT	16
2.8. @CHANGE-PASSWORD	16

Section	Page
2.9. Dialed-up Terminals	16
2.9.1. Acoustic Coupling	17
2.9.2. Wired Coupling	17
2.10. User Programs in SINTRAN III	18
2.10.1. Monitor Calls from FORTRAN Programs	18
2.10.2. Monitor Calls from MAC Assembly Programs	19
2.10.3. Monitor Call Formats	19
2.10.4. Aborting a User Program	20
3. The File System	23
3.1. General	23
3.2. Files	26
3.2.1. General	26
3.2.2. File Names	27
3.2.3. File Types	29
3.2.4. File Versions	30
3.2.5. Terminal Types	30
3.2.6. Scratch Files	30
3.2.7. Logical Device Number	31
3.2.8. FORTRAN Unit Number	33
3.2.9. Logical Device Numbers for Terminals	34
3.3. File Directories	34
3.3.1. General	34
3.3.2. Directory Commands	35
2.1. @CREATE-DIRECTORY	35
2.2. @ENTER-DIRECTORY	35
2.3. @CPEATE-USER	36
2.4. @GIVE-USER-SPACE	36
2.5. @RELEASE-DIRECTORY	36
3.4. File Creation and Deletion	36
3.4.1. @CREATE-FILE	37
3.4.2. @EXPAND-FILE	37
3.4.3. @CREATE-NEW-VERSION	38
3.4.4. @ALLOCATE-FILE	38
3.4.5. @ALLOCATE-NEW-VERSIONS	39
3.4.6. @RENAME-FILE	39
3.4.7. @DELETE-FILE	39
3.4.8. @DELETE-USERS-FILES	39
3.4.9. @SET-TEMPORARY-FILE	40
3.4.10. MDLFI (MON 54)	40
3.5. File Access Permission and Reservation	41
3.5.1. User Types	41
3.5.2. Granting Access to Files	42
3.5.3. @SET-FILE-ACCESS	42

Section	Page
3.5.4. @SET-DEFAULT-FILE-ACCESS	43
3.5.5. @CREATE-FRIEND	43
3.5.6. @DELETE-FRIEND	44
3.5.7. @SET-FRIEND-ACCESS	45
3.6. Requesting Access to Files	46
3.7. Reserving and Releasing Files	46
3.7.1. @RESERVE-FILE	47
3.7.2. @RELEASE-FILE	47
3.7.3. @RESERVE-DEVICE-UNIT	47
3.7.4. @RELEASE-DEVICE-UNIT	47
3.7.5. RESRV (MON 122)	48
3.7.6. RELES (MON 123)	48
3.8. Reading and Writing Data	49
3.8.1. General	49
3.8.2. Opening Files	49
2.1. @OPEN-FILE	49
2.2. @CONNECT-FILE	50
2.3. @SCRATCH-OPEN	50
2.4. @SET-PERMANENT-OPENED	50
2.5. OPEN (MON 50)	51
3.8.3. Sequential Input/Output	51
3.1. INBT (MON 1, FORTRAN call: INCH)	51
3.2. OUTBT (MON 2, FORTRAN call: OUTCH)	52
3.3. INSTR (MON 161)	52
3.4. OUTST (MON 162)	52
3.5. MSG (MON 32)	52
3.6. IOUT (MON 35)	53
3.7. NOWT (MON 36)	53
3.8.4. Random Input/Output	54
4.1. @RFILE	54
4.2. @WFILE	55
4.3. RFILE (MON 117)	55
4.4. WFILE (MON 120)	55
3.8.5. Closing Files	56
5.1. @CLOSE-FILE	56
5.2. CLOSE (MON 43)	56
3.8.6. Data Access Parameters	56
6.1. @SET-BYTE-POINTER	57
6.2. @SET-BLOCK-SIZE	57
6.3. @SET-BLOCK-POINTER	57
6.4. SETBT (MON 74)	58
6.5. REABT (MON 75)	58
6.6. SMAX (MON 73)	58
6.7. RMAX (MON 62)	58
6.8. SETBS (MON 76)	58
6.9. SETBL (MON 77)	59
6.10. CIBUF (MON 13)	59
6.11. COBUF (MON 14)	59
6.12. ISIZE (MON 66)	59
6.13. OSIZE (MON 67)	59

Section	Page
3.9. Commands for Copying Files	60
3.9.1. @COPY	60
3.9.2. @COPY-FILE	60
3.10. Commands for Listing File System Information	60
3.10.1. @LIST-FILES	61
3.10.2. @FILE-STATISTICS	61
3.10.3. @LIST-OPENED-FILES	62
3.10.4. @LIST-DIRECTORIES-ENTERED	62
3.10.5. @DIRECTORY-STATISTICS	62
3.10.6. @LIST-USERS	63
3.10.7. @USER-STATISTICS	63
3.10.8. @LIST-FRIENDS	63
3.10.9. @LIST-VOLUME	64
3.10.10. @WHERE-IS-FILE	64
3.10.11. @LIST-DEVICE	64
3.11. Initialising a Floppy-Disk	64
3.12. Manipulating Magnetic Tapes, Floppy Disks, Versatec and cassettes	66
3.13. Taking Backups	67
4. Sintran Information	71
4.1. General	71
4.2. Miscellaneous	71
4.2.1. @LIST-REENTRANT	71
4.2.2. @LIST-TITLE	71
4.2.3. @HELP	71
4.2.4. @TERMINAL-STATUS	72
4.2.5. @WHO-IS-ON	72
4.3. The SINTRAN III Calendar	72
4.3.1. @DATCL	72
4.3.2. @TIME-USED	72
4.3.3. CLOCK (MON 113)	72
4.3.4. TIME (MON 11)	72
4.3.5. TUSED (MON 114)	73
5. Executing User Programs and Subsystems	75
5.1. General	75

Section	Page
5.2. Creating a PROG-File	75
5.2.1. @MEMORY	75
5.2.2. @DUMP	76
5.3. Controlling Execution	76
5.3.1. @RECOVER	76
5.3.2. @CONTINUE	77
5.3.3. @GOTO-USER	77
5.4. Examining User's Registers and Memory	78
5.4.1. @STATUS	78
5.4.2. @LOOK-AT	78
5.4.3. @SET-MEMORY-CONTENTS	79
5.5. Loading Binary Programs	79
5.5.1. @LOAD-BINARY	79
5.5.2. @PLACE-BINARY	79
5.6. Error Messages from User Programs	79
5.6.1. ERMSG (MON 64)	80
5.6.2. QERMS (MON 65)	80
5.7. Communicating with One's Own Terminal	80
5.7.1. SETCM (MON 12)	80
5.7.2. COMND (MON 70)	81
5.7.3. Terminating Programs	81
5.7.4. LEAVE (MON 0)	81
5.7.5. RTEXT (MON 134)	81
6. Terminal Characteristics	83
6.1. General	83
6.2. Terminal Communication	83
6.2.1. @TERMINAL-MODE	83
6.2.2. @DISABLE-ESCAPE-FUNCTION	84
6.2.3. @ENABLE-ESCAPE-FUNCTION	84
6.2.4. TERMO (MON 52)	84
6.2.5. DESCF (MON 71)	85
6.2.6. EESCF (MON 72)	85
6.2.7. ECHOM (MON 3)	85
6.2.8. BRKM (MON 4)	85
6.3. Terminal Identification	86
6.3.1. @SET-TERMINAL-TYPE	86
6.3.2. @GET-TERMINAL-TYPE	86
6.3.3. MSTTY (MON 17)	86
6.3.4. MGTTY (MON 16)	86
6.4. Suspending the Terminal	87

Section	Page
6.4.1. @HOLD	87
6.4.2. HOLD (MON 104)	87
7. Batch- and Mode-Jobs	89
7.1. General	89
7.2. Definitions	89
7.3. Sample Batch File	90
7.4. Commands for Running Batch Jobs	91
7.4.1. @APPEND-BATCH	91
7.4.2. @ABORT-JOB	91
7.4.3. @DELETE-BATCH-QUEUE-ENTRY	92
7.4.4. @LIST-BATCH-QUEUE	92
7.4.5. @LIST-BATCH-PROCESS	92
7.5. Commands for Running Mode-Jobs	93
7.5.1. @MODE	93
7.6. Commands within Mode- or Batch-Jobs	93
7.6.1. @ENTER	94
7.6.2. @SCHEDULE	94
7.6.3. @CC	94
7.7. Monitor Calls for Mode- and Batch-Jobs	94
7.7.1. RSIO (MON 143)	94
8. Spooling	95
8.1. General	95
8.2. Appending and Changing the Queue Entry	98
8.2.1. @APPEND-SPOOLING-FILE	98
8.2.2. @SET-NUMBER-OF-PRINT-COPIES	98
8.2.3. @DEFINE-SPOOLING-FILE-MESSAGE	99
8.3. Changing the Order in the Queue	99
8.3.1. @MOVE-SPOOLING-QUEUE-ENTRY	99
8.3.2. @REMOVE-FROM-SPOOLING-QUEUE	99
8.3.3. @DELETE-SPOOLING-FILE	100
8.4. Starting and Stopping the Print	100
8.4.1. @START-PRINT	100
8.4.2. @STOP-PRINT	100
8.4.3. @RESTART-PRINT	100

Section	Page
8.4.4. @ABORT-PRINT	100
8.5. Adjusting the Print	101
8.5.1. @FORWARD-SPACE-PRINT	101
8.5.2. @BACKSPACE-PRINT	101
8.6. Statistics	101
8.6.1. @LIST-SPOOLING-QUEUE	101
8.6.2. @SPOOLING-PAGES-LEFT	102
8.6.3. @LIST-SPOOLING-FORM	102
8.7. Monitor Calls for Spooling	102
8.7.1. SPCLO (MON 40)	102
8.7.2. RSPQE (MON 55)	103
9. Sending Messages to Other Terminals	105
9.1. @MAIL	105
9.2. @OPERATOR	105
9.3. @WAIT-FOR-OPERATOR	106
10. NORD-NET	107
10.1. Introduction	107
10.2. The Communication Line	108
10.2.1. General	108
10.2.2. @COMMUNICATION-STATUS	109
10.2.3. @COMMUNICATION-LINE-STATUS	109
10.3. Remote Terminals	110
10.3.1. General	110
10.3.2. @REMOTE	111
10.3.3. @LOCAL	112
10.3.4. Example of @REMOTE and @LOCAL	112
10.3.5. Detailed Description of Remote Terminal Connection	114
10.4. Remote File Access	115
10.5. Data Transfer	117
10.5.1. General	117
10.5.2. WRQI (MON 163)	121

<u>Section</u>	<u>Page</u>
APPENDIX A, Editing Control Characters for Commands	123
APPENDIX B, Index	127

TABLE OF FIGURES

<u>Fig nr</u>		<u>Sec nr</u>
1	A User CRT Terminal	2.1
2	A Console Terminal (Hard-Copy)	2.2
3	The Log In Procedure	2.2
4	The States of a User Terminal	2.3
5	Categories of Hardware Devices	3.1
6	Data Organisation on a Mass Storage Medium	3.1
7	Indexed File	3.2.1
8	Contiguous File	3.2.1
9	Allocated File	3.2.1
10	File Types	3.2.3
11	Categories of Logical Devices	3.2.5
12	Normal File Protection	3.5.1
13	Granting Access to Files	3.5.6
14	Spooling System for Time sharing users	8.1
15	Spooling System, Queue Entry	8.1
16	A Communication Line	10.1
17	Example of Logical Device Numbers in NORD-NET	10.2.1
18	Remote Terminal Processing	10.3.1
19	Remote Processing, State Diagram	10.3.4
20	Remote Program Using a Local File	10.3.5

1. Introduction

1.1. General

An ND computer system is a medium scale, general purpose computer system which, because of its modular design, can be configured for a wide variety of applications.

SINTRAN III is a multiprogramming, multi-lingual operating system that supervises the processing of real-time, time-sharing, and batch programs submitted to an ND computer system. SINTRAN III relieves the user from much of the mundane work connected with the developing and running of user programs.

A SINTRAN III system can run in as little as 32 K words, the memory resident SINTRAN III/RT. The virtual storage versions, SINTRAN III/VSE and VSE-500 may be expanded with up to 32 Mbytes of main memory, up to eight 288 MB disks, connections to other computers, etc. Several ND computers may therefore form a computer network.

This manual is primarily oriented towards the SINTRAN III/VS systems. The SINTRAN III/RT system is documented in "SINTRAN III - RT USER'S GUIDE (ND-60.082)".

Each terminal and batch activity is run as a separate process. These processes are called background programs and are run on a time sharing basis in the ND computer. The other type of processes, foreground programs, are created by real-time programmers. They normally run with a higher priority in the computer. In general, a real-time program may be either a background or foreground program.

1.2. User Categories

While this manual is only concerned with time-sharing users, other types of users recognised in SINTRAN III are the real-time user, developing foreground programs, and system supervisors, managing the computer installation. Another type of user is the "end user", performing application oriented tasks such as data entry or data inquiry through application programs. This user will not be discussed in the SINTRAN III documentation.

1.2.1. Time-Sharing and Batch Users

This user group will access the system through a terminal or by submitting jobs for batch processing from disk files or card decks containing the necessary commands and data.

Time-sharing and batch users are known to the system by names which must be presented when logging in as timesharing users or as a parameter to the first command in a batch job. The user name may be protected by a password chosen by the user to prevent unauthorized

access to the system.

The user name must be created by those responsible for the system. Time-sharing and batch users access the system in its "background" processing mode, normally executing programs with a lower priority than real-time programs executing in "foreground" mode. The time-sharing and batch processes are normally given processing time on an equal basis.

Trying to execute real-time user and system supervisor commands is not permitted and will produce an error message.

1.2.2. The Real-Time User

The operating system recognizes a special user name RT as a privileged user capable of controlling real-time programs. This user name should be protected by a password to prevent unauthorized access.

This user is permitted to use commands for developing and running real-time programs. The manual "SINTRAN III REAL-TIME GUIDE" (ND-60.133) gives the details about this activity.

The user RT may also use the time-sharing and batch facilities for program development and testing purposes. Trying to execute system supervisor commands is not permitted and will produce an error message.

1.2.3. System Supervisor

The operating system recognizes a special user name SYSTEM as the privileged user having the access to all commands in SINTRAN III. The special SYSTEM commands facilitate creating and deleting users, creating and deleting directories, together with other commands for system management. The manual "SINTRAN III SYSTEM SUPERVISOR" (ND-60.103) gives the details about system management.

This user name should be protected by a password.

1.3. Commands and Monitor Calls

A command may be entered from a terminal, a mode input file, or a batch input file. The command is checked for validity and the appropriate function is performed.

Commands are classified according to user categories.

Time-Sharing/Batch Commands.

- calling compilers, assemblers, editors, other subsystems and user programs
- creating, deleting and maintaining files

- reserving and releasing files, devices, etc.
- communication with remote computers.

Real-Time Commands

- starting and stopping RT programs
- loading new RT programs using the RT loader
- reserving and releasing peripheral devices on behalf of RT programs, etc.

The Real Time Guide (ND-60.133) describes these functions.

System-Supervisor Commands

- starting and stopping the system
- creating, entering and releasing file directories on disk
- creating and deleting users
- allocating resources, etc.

The System Supervisor manual contains a guide to using these functions.

Monitor Calls

A monitor call is a special instruction in machine code. It is used to call service functions within the operating system. Monitor calls in machine code or assembly code can only be performed in MAC. In high level code such as FORTRAN, etc. a corresponding set of subroutine calls is used. In this manual, the examples show only the FORTRAN call, if it is available.

1.4. Hardware Environment

The minimum hardware configuration required to run SINTRAN III/VS is:

- NORD-10/S or ND-100 with memory management system
- main memory of 128 Kbytes
- disk unit with controller
- floppy disk unit
- console terminal

SINTRAN III/VSE requires an ND-100, while SINTRAN III/VSE-500 requires an ND-500 system.

The range of standard peripherals include alphanumeric, graphic, and colour display systems, hard-copy terminals, printers, plotters, paper tape readers and punches, card readers and punches, magnetic tape stations, floppy disks, hard disks, A/D and D/A convertors, telex, modem, CAMAC interfaces, etc. Processor options includes local and multiport memory up to 32 Mbytes, Direct Memory Access and cache memory. Norsk Data marketing information gives further details about the hardware available.

1.5. SINTRAN III Subsystems

Some subsystems are integrated into the operating system. They are the file system, the RT-loader, the spooling system, the assembler/debugger for RT programs, and parts of the accounting system. The other subsystems are distinct from, but run under, the operating system. They are compilers, text editors, program loaders, data-base systems, etc. Norsk Data marketing information gives further details about these subsystems.

1.6. Documentation

All commands and monitor calls are documented in detail in the SINTRAN III REFERENCE MANUAL (ND-60.128). That manual contains complete definitions listed in alphabetical order.

The SINTRAN III user's guides, on the other hand, document SINTRAN III facilities in a functional order and do not contain all details. The time-sharing/batch guide describes most of the facilities available to these types of users. Some commands and monitor calls are restricted to real-time users only. They are described in the Real Time Guide. Special facilities available to the user SYSTEM are described in the System Supervisor manual.

In addition there are two reference manuals for special subsystems under SINTRAN III. The RT Loader manual contains reference documentation for this subsystem alone, while the SINTRAN III Utilities manual documents the BACKUP-SYSTEM, MAIL, ACCOUNTING, LOOK-FILE, FILE EXTRACT, PERFORM and GPM subsystems.

2. Using The Terminal

2.1. The ND Terminals

A User CRT terminal (figure 1) is used for the interactive communication with the system. All terminals connected to a system can function as user terminals. However, one terminal is always selected as the error device. This is the terminal where the system error messages will appear.

Each ND computer system contains a console terminal as shown in figure 2. It is first used by the System Supervisor to get SINTRAN III operational. After this is done, the console terminal will normally function as the error device. However, the System Supervisor can select any terminal to be the error device. So, a console terminal is always the same physical terminal. However, being an error device is a function that can be assigned to any output device.

Appendix A of SINTRAN III REFERENCE MANUAL (ND-60.128) shows the standard terminals delivered with ND systems. If your terminal is not documented here, refer to the manufacturer's "User's Manual" for definition of the keyboard layout, or contact your local ND representative.

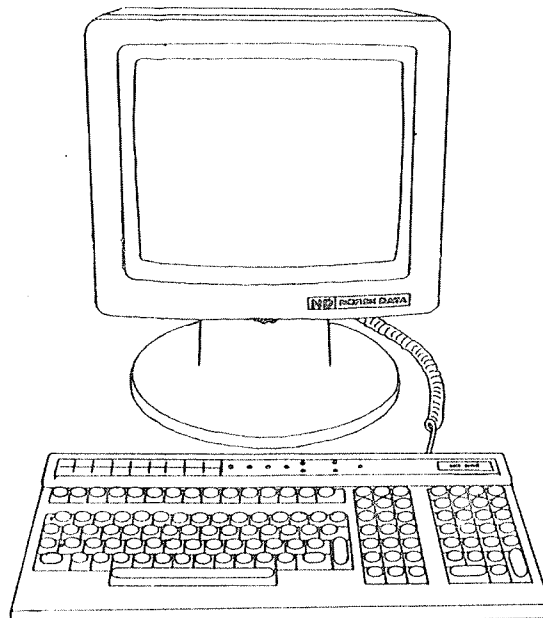


Fig. 1. A User CRT Terminal

2.2. Logging In

You are probably familiar with the process of logging in by now, but in this section we give a more general explanation of the procedure.

The System Supervisor is responsible for configuring the terminals (i.e., cabling, selecting speed of transmission, etc.). The rest of the process involves the user.

After turning on the terminal and making sure the terminal is online (press LINE if necessary), press ESCAPE and SINTRAN III returns the time and asks for the user's identification("User Name") created for him by user SYSTEM. When this user name has been entered, followed by CR, a prompt is received for a password. This is optional and can be any set of characters terminated by CR. If used, the password will be checked but not echoed for security reasons. If it is not used, only CR should be entered. If the accounting system is running a further prompt is now issued for a project password. This is also optional, its function being to identify responsibility for the cost of the computer time. The password, if any, followed by CR, should be entered, whereupon the logging-in procedure is complete and the commercial-at (@) will be displayed indicating that SINTRAN III is waiting for a command or call to a subsystem.

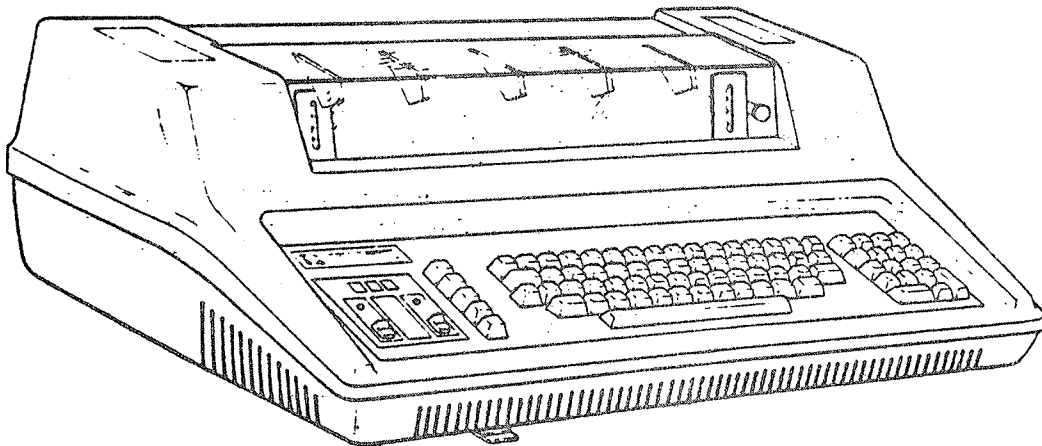


Fig. 2. A Console Terminal (Hard-copy)

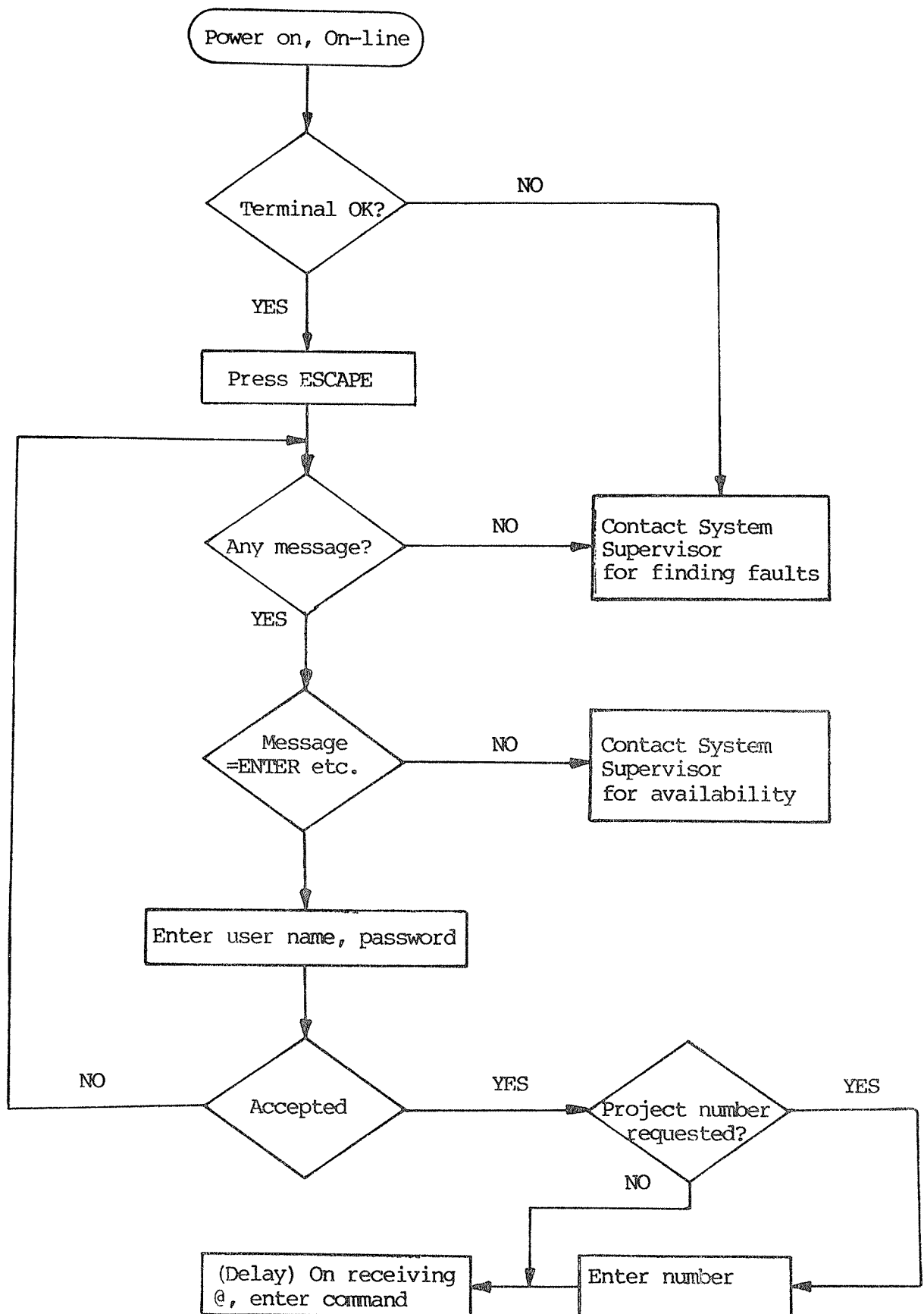
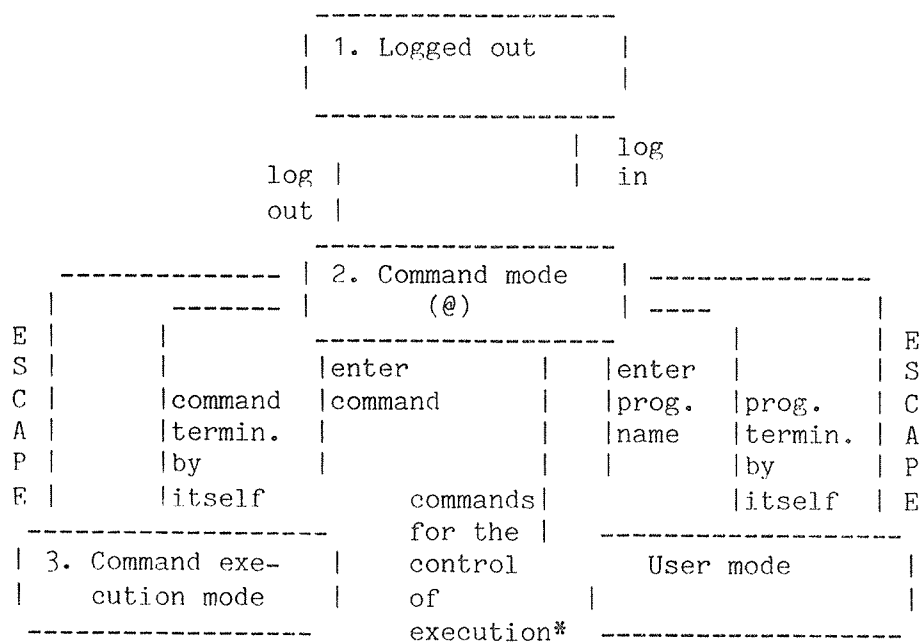


Fig. 3. The Log In Procedure

2.3. Executing Commands and Programs

The different states of the interactive terminal work are shown in figure 4. After you have logged in, you enter the command mode as indicated by the @ on the screen. The terminal is now waiting for a command or the name of a program. If you enter a command (consisting of name and parameters) you will enter the comand execution mode. The command may be written as upper or lower case letters. It may terminate by itself or you can in most cases terminate by pressing ESCAPE. In the command mode you can also enter the name of a program. This can be either a system program (i.e., subsystem, a program supplied with the machine) or a user program (a program written by the user). The program is loaded and the terminal enters the user mode. All terminal input (except ESCAPE-if enabled) is now handled by the user program. In a way similar to command execution, the program may terminate by itself or you may at any time press ESCAPE to stop the execution (if not disabled by the program). When you receive @, you are back in command mode.



* These commands are @RECOVER, @CONTINUE, and @GOTO-USER

Fig. 4. The States of a User Terminal

2.4. Writing SINTRAN III Commands

2.4.1. Single Line Commands

A SINTRAN III command is normally written on one line. As an example, consider the command below.

Definition:

@LIST-FILES <file name>,<output file>

Example:

@LIST-FILES OLE,TERMINAL

The command elements are: command name (LIST-FILES) and parameters (OLE and TERMINAL). The elements are separated by space or comma. The command name consists of one or more parts (LIST and FILES) separated by hyphen (-). Parameters are defined in the documentation by enclosing the name in angular brackets (<file name>). A parameter may be one of four types;

1) A numeric parameter is either octal or decimal. If you specify just the digits, the radix is determined by the command. The radix can be specified explicitly by appending a B for octal or D for decimal, e.g.:

@GET-TERMINAL-TYPE 35

will use 35 as a decimal number. The command could have been written as:

@GET-TERMINAL-TYPE 43B

2) A parameter may be a file name. This is either the name of a data file or a peripheral file.

3) The parameter can be a selection, i.e., an answer to a question, such as YES or NO.

4) The parameter can be a character string terminated by an apostrophe (') or CTRL/L before the comma or space.

2.4.2. Abbreviation in Commands

Command names, file names and selections can be abbreviated. Command names can be abbreviated as long as they are unique, e.g., @LIST-FILES may be written as @L-FIL or @LI-FI. However, @LI-F would give the message:

AMBIGUOUS COMMAND

since it can mean both @LIST-FILES and @LIST-FRIENDS. The same applies to file names and selections. When creating a file, however, the name must be given in full.

Sometimes we want to refer to several objects with one abbreviation. For example, the command:

@LIST-FILES OLE,TERM

will list all files matching the name OLE, such as OLE-1, OLE-2, OLEANNA, etc.

2.4.3. Use of Asterisk (*)

If an asterisk (*) appears in a command element, it will match any character, e.g.:

@LIST-FILES *COPY,TERM

will list files such as ACOPY, BCOPY etc.

2.4.4. Default Values

Many parameters have default values. If the parameter is omitted (as indicated by two subsequent element separators), the default value will be selected, e.g.:

@LIST-FILES,,LINE-PRINTER

will list all the user's own files in his/her default directory on the printer. (Default directory will be explained in chapter 3.) The command:

@LIST-FILES OLE,,

will list all files matching the file OLE on the TERMINAL, which is the default <output file>. Note that two commas must end this command or else the last parameter will be asked for.

2.4.5. Prompting Line Commands

Until now, all command elements have been entered on the same line. It is also possible to enter each command element on a separate line:

```
@LIST-FILES  
FILE NAME:OLE  
OUTPUT FILE:TERMINAL
```

Each entry is followed by RETURN. Leading text requesting parameters are called prompting lines. Just pressing RETURN as an answer to a request indicates a default value, e.g.:

```
@LIST-FILES  
FILE NAME:  
OUTPUT FILE:
```

has the same function as:

```
@LIST-FILES,,,
```

If you do not specify all parameters in a command, the rest will be asked for (prompted). For example:

```
@LIST-FILES OLE,,  
OUTPUT FILE:
```

The last parameter is prompted, since the first line ended with only two commas.

2.4.6. Correcting Typing Errors

To correct errors while you are typing, SINTRAN III has control characters available. They are the same characters used by the EDIT command in the QED subsystem. You can correct the line you are currently typing, or edit the previous command and enter it again. In the latter case, the previous command must have been a single line command. This section demonstrates a minimum set of control characters. For a more complete list, refer to appendix A.

2.4.6.1. Correcting the Current Line

CTRL/A deletes the previous character in the current line or effectively backspaces the line. SINTRAN responds with a ^ (up-arrow). Assume the cursor is positioned after the last character in the (erroneous) line,

@TERMINAL-STATUS

to eliminate the last U, type CTRL/A twice and resume the typing from the error.

@TERMINAL-STATUS^^S,,,

The up-arrows are echoes to CTRL/A and indicates that you have eliminated the previous two characters.

CTRL/Q deletes the current line. If you have typed the following and the cursor is positioned after the U:

@TARMINAL-STATU

you might prefer to start the line all over again. Hit CTRL/Q and the cursor will advance to the beginning of the next line where the command can be typed again. SINTRAN responds with underline

@TARMINAL-STATU
TERMINAL-STATUS

2.4.6.2. Editing the Previous Line

The basic commands for this purpose are CTRL/D, CTRL/C, CTRL/S, CTRL/E, CTRL/Z and CTRL/O.

CTRL/D copies the rest of the old command (on the previous line) to the new command (on the current line). You can use it to enter the previous single line command once again. Or, you can use it to terminate a correction. If by mistake you type:

@TARMINAL-STATUS,,
NO SUCH FILE NAME
@

you can correct it by typing T, E, and CTRL/D,

@TERMINAL-STATUS,,

...
The command has now been correctly entered.

CTRL/C copies one character from the old command to the new command. For example:

@LU-FI,,,
NO SUCH FILE NAME
@

Now, type CTRL/C once, then the correct character (I instead of U) and then CTRL/D to terminate the command.

@LI-FI,,,

Your command will now be accepted.

CTRL/S skips one character in the old command, e.g.:

```
@LII-FI,,,  
NO SUCH FILE NAME  
@
```

To correct the command, enter CTRL/C twice, then CTRL/S and finally CTRL/D.

```
@LI%-FI,,,
```

When CTRL/S is typed, SINTRAN responds with percent (%) instead of the character in the old command.

CTRL/E is used for insert mode. The first time CTRL/E is typed you enter insert mode, the next time you type CTRL/E you leave insert mode. In insert mode, all printing characters will be inserted in the old command. For example:

```
@L-FI,,,  
AMBIGUOUS COMMAND  
@
```

Press CTRL/C, CTRL/E and enter the missing character. Finish the editing with CTRL/E and CTRL/D.

```
@L<I>-FI,,,
```

When you enter insert mode a start angle bracket (<) is echoed. When you return from this mode an end angle bracket (>) is echoed. There is no restriction on how many times you may use CTRL/E on one line.

2.5. Error Messages from SINTRAN III

This section gives an overview of the error messages you may get at the terminal. A complete list of error messages is included in appendix D of the SINTRAN III REFERENCE MANUAL.

There are three types of error messages:

a. Run-time errors are errors directly from the SINTRAN III kernel. (I.e., the central part of the operating system.) The message format is as follows:

```
ERROR nnn AT aaaaaa; <error text>
```

The meaning of the number nn and the <error text> is shown in Appendix D.1.2 of the SINTRAN III REFERENCE MANUAL which lists the different error messages.

b. File-system errors are displayed only as <error text>. The different texts are shown in the above-mentioned Appendix under the column "meaning".

c. Command processor errors are displayed as <error text>.

Errors may also occur when you are running user programs. Error messages are then either displayed directly by SINTRAN III on the terminal, or returned with an error indicator from a monitor call.

a. Run-time errors may occur in the same way as for commands. The message is displayed directly on the terminal.

b. File-system errors are indicated on return from monitor calls. For example, OPEN (MON 50) has an error exit with the A-register containing the error number. The numbers are listed in appendix D.2.1 of the SINTRAN III REFERENCE MANUAL. The monitor calls ERMSG (MON 64) or QERMS (MON 65) are normally used to display the <error text> on the user terminal. A call may then look as follows:

```
MON 50          %OPEN
MON 65          %QERMS, TERMINATE ON RETURN
<normal return>
```

c. Command-processor errors may also be caused by monitor calls. The <error text> is displayed. The messages cannot be produced by ERMSG or QERMS.

d. FORTRAN run-time errors are written out directly from a FORTRAN library routine according to the format:

```
FORTRAN RUN-TIME ERROR nnn AT ADDRESS aaaaaa
<error text>
```

The number nnn and the corresponding <error text> is shown in appendix F.1 of the SINTRAN III REFERENCE MANUAL.

Errors may occur in SINTRAN III subsystems. These system programs have their own set of messages. Consult the user manual of the subsystem for documentation of error messages.

Finally, error messages from Real-Time programs are routed to the error device (normally the console terminal.) This also applies to input/output messages from the batch output device.

2.6. Listing Information about the System

At this point you should have become familiar with the way SINTRAN III commands are written. To exercise your skill, you can try out some commands which list information about the system. The first group lists information about the other users. Each command below is explained together with its definition and an example.

@LIST-USERS <user name>,<output file>
@LIST-USERS,,,

This command lists all users defined in main and default directories in the file system. The list appears on the TERMINAL and it will be similar to the one below:

```
USER 0 : PACK-ONE:SYSTEM
USER 1 : PACK-ONE:GUEST
USER 2 : PACK-ONE:TPS-SYSGEN
USER 3 : PACK-ONE:HAP-SB
USER 4 : PACK-ONE:BACK-SIBAS
USER 5 : PACK-ONE:ACEM-BOH
USER 6 : PACK-ONE:JOHN-SMITH
USER 7 : PACK-ONE:JOE-GREEN
USER 8 : PACK-ONE:FLOPPY-USER
USER 9 : PACK-ONE:JERRY-JONES
```

@WHO-IS-ON

This command will show which users are logged in on time sharing terminals, batch processors or NORD-NET connections. It will be a subset of the previous list and might look as follows:

```
      1  SYSTEM
      35 LASSE
==> 48  GUEST
      670 SYSTEM
```

The first column shows the logical device number (see section 3.1) of the terminal where the user is logged in. The arrow indicates your own terminal.

@TERMINAL-STATUS <logical dev. no.>,<interval>
@TERMINAL-STATUS,,10

LOG.NO	USER	MODE	CPU-MIN	OUT OF	LAST COMMAND
1	SYSTEM	COMMAND	0	15	SYS
35	LASSE	USER	0	50	CC <TED> CC
48	GUEST	COMMAND	4	54	FTN

It is possible to obtain a periodic report on the activity of one or all terminals logged in. The above example produces a report on all terminals every 10 seconds. The reporting will continue until you press ESCAPE to get back to command mode.

The last set of commands is used to check the time.

@DATCL

The command gives you the date and time according to SINTRAN III clock.

@TIME-USED

The command lists the amount of CPU time you have used and the period for which you have been connected (called the connect time).

2.7. @LOGOUT

When you want to finish using the terminal, you log out from the system by giving the command:

```
@LOGOUT                                (or just @LOG)
10.09.18      28 JULY    1982
--EXIT--
```

If the accounting system is running, the "TIME-USED" information is also displayed.

```
@LOGOUT
10.09.18      28 JULY    1982
TIME USED IS   1 MINS   23 SECS OUT OF   4 MINS   48 SECS
--EXIT--
```

At this point you are no longer logged in on the computer.

It is good practise to log out if you are leaving the terminal for more than a few minutes.

2.8. @CHANGE-PASSWORD

@CHANGE-PASSWORD <old password>,<new password>

The password can be changed at any time. If you do not want to have the password displayed on the screen, enter the command as multiple lines, e.g.:

```
@CHANGE-PASSWORD
OLD PASSWORD:      (old password is entered but is not echoed)
NEW PASSWORD:      (new password is entered but is not echoed)
@
```

2.9. Dialed-up Terminals

Some users access the system through dialed-up telephone lines. These users must establish phone connection with the computer in addition to the normal log-in procedure. When logging out the phone must be disconnected.

Most terminals work in the full-duplex mode. This means that you can send data to the computer at the same time as the computer is sending data back, just as when you use a normal terminal connection.

There are two types of couplings at the terminal site, acoustic coupling and wired coupling.

2.9.1. Acoustic Coupling

A terminal with this type of coupling has two rubber cups where the telephone receiver is mounted. Logging in proceeds as follows:

1. Turn the terminal on.
2. Dial the computer. You should get high pitch tone.
3. Mount receiver in acoustic coupler.
4. Press ESCAPE to start the normal log-in procedure.

The log-out procedure is as follows:

1. Enter the @LOGOUT command.
2. Put the receiver back on the hook to disconnect the line.

2.9.2. Wired Coupling

A terminal with this type of coupling is wired to the telephone modem. Logging in proceeds as follows:

1. Turn the terminal on.
2. Turn switch on modem to the "PHONE" position.
3. Dial the computer to get the high pitch tone.
4. Turn switch to "DATA" position and put receiver back on the hook.
5. Press ESCAPE to start the normal log-in procedure.

The log out procedure is as follows:

1. Enter the @LOGOUT command.
2. Turn switch to the "PHONE" position.
3. Lift the receiver momentarily from the hook to ensure that the line is disconnected.

2.10. User Programs in SINTRAN III2.10.1. Monitor Calls from FORTRAN Programs

The previous sections have described various commands the user can enter at the terminal. When running programs, it is possible to use the facilities of SINTRAN III by means of monitor calls. In FORTRAN the call is made to a subroutine or function. The routine then checks the parameters and makes the corresponding monitor call in machine code.

A small program to demonstrate the use of FORTRAN monitor call is shown below. It obtains the time used since you logged in by calling TUSED. Firstly we use QED to enter the symbolic code. (For a description of QED, see the QED USER'S MANUAL (ND-60.031)).

```
@QED
QED 4.3
*A
      DOUBLE INTEGER TD,TDD,TUSED
      TD=TUSED(I)
      TDD=TD/50
      WRITE(1,10)TDD
10    FORMAT(' TIME USED IS ',I5,' SECONDS')
      STOP
      END
                                     (Type RETURN and CTRL/L)
*W "TEST"
64 WORDS WRITTEN
*EX
@
```

The program file created is TEST:SYMB. It can now be compiled by the FORTRAN compiler as follows:

```
@FTN
NORD 10/100 FORTRAN COMPILER FTN-2090H
$COM TEST,,"TEST"
7 LINES COMPILED . OCTAL SIZE = 130
CPU-TIME USED IS 0.3 SEC.
$EX
@
```

The input to the compiler was the file TEST:SYMB (SYMB=default) and a new file of object code, TEST:BRF (BRF=default), was created. The program is ready to be loaded and run by the NRL, ND Relocating Loader:

```
@NRL
RELOCATING LOADER LDR-1935H
*LOAD TEST,FTMLIB
FREE : 035116-177777
*RUN
```

TIME USED IS 3 SECONDS

022202 STOP 0
@

2.10.2. Monitor Calls from MAC Assembly Programs

Monitor calls are made directly in MAC programs by using the MON n instruction, where n is the monitor call number. A program similar to the one in the previous section is shown below. The monitor call for TUSED is MON 114. IOUT (MON 35) is used to print the number.

In order to enter the program use is made of the direct program entry feature of MAC.

```
@MAC
- MAC -
10/xyzpqr MON 114      set location counter to 10, call TUSED
SAT 62                divide AD by 50, AD contains time used in
RDIV ST               basic time units, i.e. 20msec. Division
                        gives quotient in A, remainder in D.
SAT 12                print in decimal
MON 35                ...
MON 0                 exit to command mode
10!                   start execution in loc. 10
@
```

In this case, we have used 2 seconds of CPU time. The result can be verified by using the command corresponding to TUSED:

@TIME-USED

TIME USED IS 2 SECS OUT OF 9 MINS 44 SECS

@

2.10.3. Monitor Call Formats

The monitor calls described in the previous two sections are all available from time-sharing and batch programs. Some calls are only available from RT-programs. Refer to the SINTRAN III REFERENCE MANUAL for the availability of each monitor call.

In general, passing parameters in MAC monitor calls is performed in two different ways:

A: Through hardware registers. Normally the T, A, D and X registers are used. The input parameters must be loaded by the user program before the call is made. An example is INBT (MON 1),

SAT 1	%device no. is 1
MON 1	%INBT - read one character
JMP ERR	%error return, A = error no.
STA CHR	%skip return, A = character in bits 7-0

B: By using a standard call format. The A register points to a parameter list containing the addresses of the values of each parameter, or the values themselves, or a mixture of the two. An example is CLOCK (MON 113)

LDA (PAR	%A reg. points to parameter list
MON 113	%CLOCK - read date and time
...	% next instruction
)FILL	
PAR, ARRAY	%pointer to 7 word array
...	
ARRAY,0	%basic units
0	%seconds
0	%minutes
0	%hours
0	%day
0	%month
0	%year

Most monitor calls use the two following instructions as return points (see example A above), the first if an error occurs, the second when the function is performed correctly. In case of the error return, the A register usually contains the file-system error number. Appendix D.2.1 of the SINTRAN III REFERENCE MANUAL lists the different errors.

When executing a MON instruction within a user program, an interrupt is generated, transferring control to a system routine on interrupt level 14. This routine activates the required function on a lower interrupt level (1, 3, 4, or 5) on behalf of the calling program.

2.10.4. Aborting a User Program

As mentioned in section 2.3, a user program or subsystem can be aborted by pressing the ESCAPE key, unless escape is disabled e.g. in a SIBAS application program. The terminal will then display the message

USER BREAK AT aaaaaa

where aaaaaa is the next address to be executed after the point of interruption. All opened files will be closed, except the ones which

are permanently opened (Section 3.8.2).

Three commands are useful after aborting a program, they are:

@STATUS

Display the contents of the program registers at the point of interruption.

@CONTINUE

Restart the program at the restart address if any (Section 5.2.2).

@GOTO-USER <address>

Restart the program at <address>. The default address is the next one (aaaaaa, as shown above). If the program uses files when it is interrupted, it is essential to set them permanently opened before the program is started for the first time. They will then still be open when the program is restarted.

3. The File System

3.1. General

The SINTRAN III File System is designed to manipulate files on disks and floppy-disks. A "file" in this context is a collection of records or blocks, ordered randomly or sequentially. It is also able to handle floppy disks, magnetic tapes and standard peripherals as sequential files. The file system is designed to operate as part of SINTRAN III.

Each file in the file system is named with a character string, and these strings are used in all commands to the file system. The file has a owner, which must be one of the users in the file system. Each user may have several (up to 8) of the other users as friends (see section 3.5.1.). The file system provides individual protection of files, with separate protection modes for the owner, the owner's friends and the public access to the file.

The structure of the physical devices connected to an ND computer is shown in figure 5. The lowest element of the hierarchy is a device or a data storage medium. (Note that this is just a logical organization of devices by functional category as opposed to how they are physically interconnected.) A peripheral file name is associated with each of the character devices and may be associated with a NORD-NET line.

A directory is associated with each of the mass storage media except for tape. The data area of a directory is divided into user areas and each area will contain a set of mass storage files (see figure 6). A file name may thus be either a peripheral file name or a mass storage file name. For example, if you want to copy a file from a mass storage file (OLE) to a peripheral file (LINE-PRINTER) you can just write:

@COPY LINE-PRINTER,OLE

where LINE-PRINTER is the name of a line-printer. If you want to copy between two mass storage files, you can use the same command:

@COPY OLE,PER

The command copies to mass storage file OLE from the mass storage file PER byte by byte. A better command for this is the COPY-FILE command described later in the manual: It is much faster and it avoids the problem with holes. The system will itself make the distinction between the two types of files.

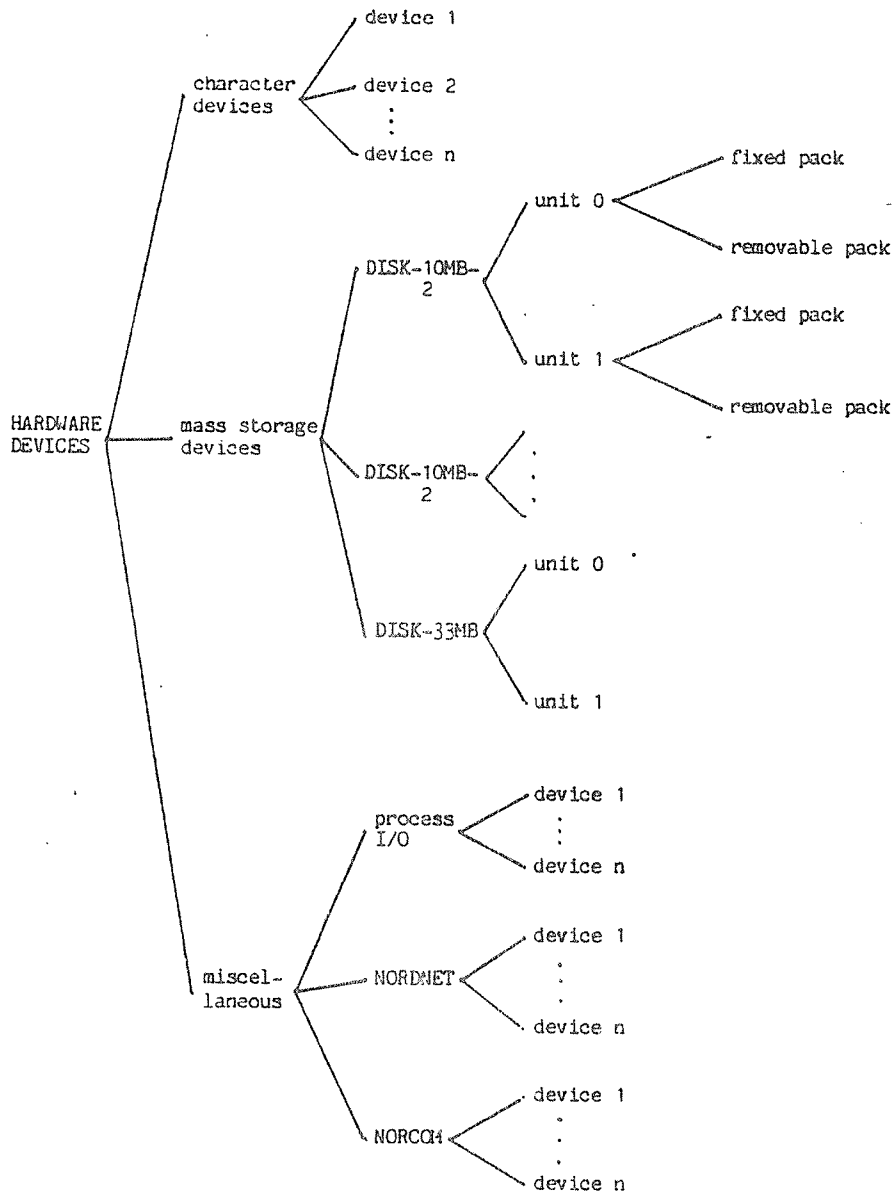


Fig. 5. Categories of Hardware Devices

Magnetic tapes behave like peripheral files to a large extent, and can be treated as a character peripheral by using INBT(MON 1) and OUTBT(MON ?). Input and output is, however, usually performed by means of the special monitor call MAGTP which is also used for all the subsidiary functions on tapes (e.g. rewind, write, and end of file). SINTRAN III Special I/O Guide, ND.60.134, describes the handling of magnetic tapes in full.

When programming in FORTRAN, magnetic tapes are treated as they are, i.e. blocks in series, thus obviating special attention. Note that Backspace, Endfile, and Rewind statements can be used to control magnetic tapes from FORTRAN.

When a file is opened for data access it is associated with a number which is then used for all subsequent access operations. Access takes place via the CONNECT command, OPEN command, or from within a program by means of the OPEN monitor call, or an OPEN statement in FORTRAN or COBOL. This number is limited and restricted to certain values. The numbers for disk files are supplied on Open and they can no longer be used after CLOSE until another OPEN or CONNECT is performed. For magnetic tape and other peripheral files the number is that of the device (see section 3.2.7).

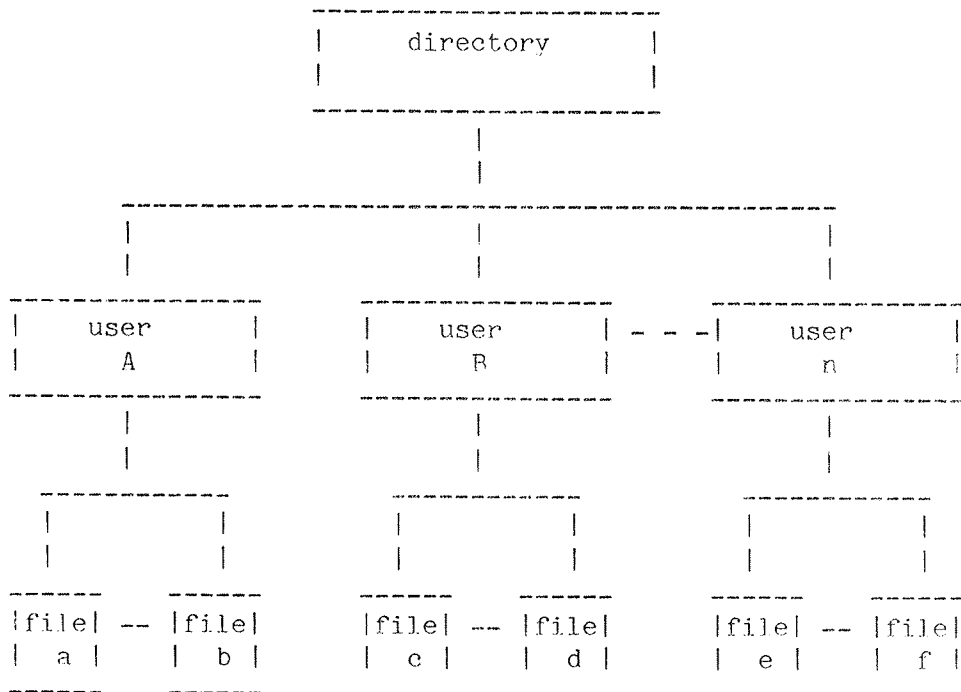


Fig. 6. Data Organisation on a Mass Storage Medium.

3.2. Files

3.2.1. General

A file is a collection of data under a given name. The term file in SINTRAN III applies both to mass storage files (figure 6) and peripheral files. A mass storage file may be, OLE:DATA. A peripheral file may be, LINE-PRINTER, the name associated with the line-printer peripheral. The smallest unit of readable data in a file is a character or byte which is always 8 bits. In mass storage files the bytes are grouped into equal sized blocks. The normal block size is 512 decimal bytes (or 256 words). The block concept is only used for direct addressing of data within the file.

The allocation of data on a mass storage device is in terms of 1K word pages. Blocks and pages are independent collections of the same data. Random reading and writing of data to a mass storage file (see below) are made in terms of blocks. A block in this connection is a collection of bytes where the length is determined by the I/O command or monitor call. FORTRAN I/O is transmitted in terms of records. The length of a record is data-defined.

Mass storage files are of two types. A type commonly used is the indexed file. Such a file may have its 1K word pages scattered around the mass storage device.

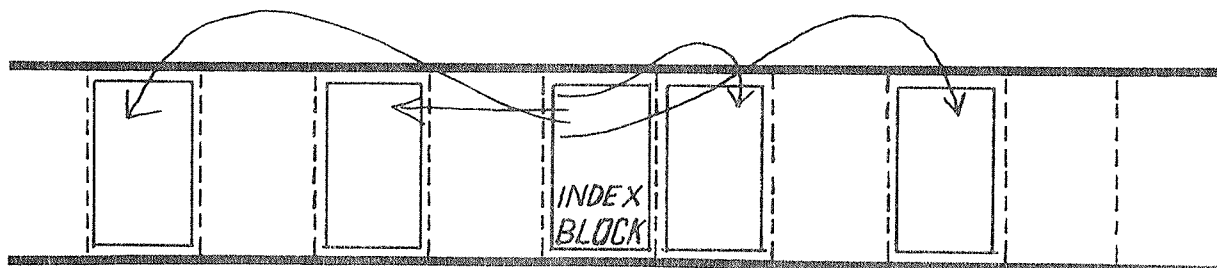


Fig. 7. Indexed File

It has an index with pointers to where the pages are located. The size of the file may expand dynamically as the user writes onto the file. The index is kept in a separate page belonging to the file so that the file always needs at least one more page than it has pages of data.

Not all pages in an indexed file need to be allocated. If a page is not written into, then the page is not allocated and the index table is then said to contain a hole.

The other file type is contiguous file. The pages of this type of file are then placed on consecutive page numbers of the mass storage device. The file has a fixed size during all data access. The advantage of a contiguous file is that data access may be faster since there is no need to look up an index.

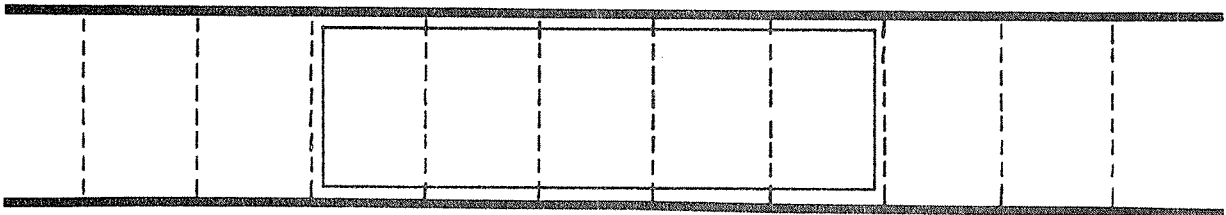


Fig. 8. Contiguous File

An allocated file is a special type of contiguous file, because here the user can specify where on the disk or floppy the pages should be placed, provided these pages are free to use.

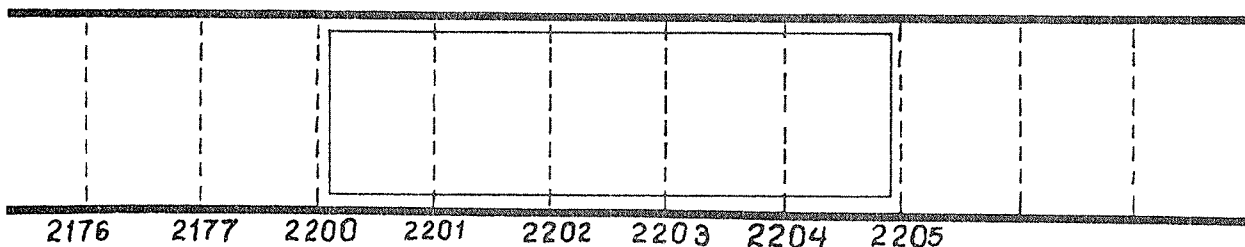


Fig. 9. Allocated File

Reading and writing to files is performed by sequential access or by random access. Sequential access implies that reading or writing starts at the first block or record and proceeds sequentially through the records in the file. Reading and Writing to random blocks in the file is called random access. All blocks in random files must be of the same size. The block size can be set by @SET-BLOCK-SIZE (SETBS (MON 76)).

3.2.2. File Names

The format of a file name is defined as follows:

(<file directory name>:<owner name>)<object name>:<type>;<version>

The permissible characters used are numeric for <version>, alphanumeric for <type> and alphanumeric and hyphen (-) for all the others. <file directory name>, <owner name> and <object name> can contain at maximum 16 characters each. <type> can at maximum contain 4 characters. The maximum value of <version number> is 256.

It is not always necessary to specify the file name in full. As an example, assume you want to create an indexed data file in your own user area on your default directory. Enter the following:

@CREATE-FILE OLE,,

In this case you need only enter the <object name>. Default values are used for the rest of the name. For the full name:

@LIST-FILES OLE,,

FILE 16 : (BIG-PACK:GUEST)OLE:DATA;1

@

returns a message similar to the one above. SINTRAN III finds that you are logged-in as user GUEST and that GUEST's default directory is BIG-PACK. File type is DATA and its version number is 1. (If there were more than one version, each one would be listed separately with its version number.)

Also, the character * will match any character. (See section 2.4.3 for further details).

3.2.3. File Types

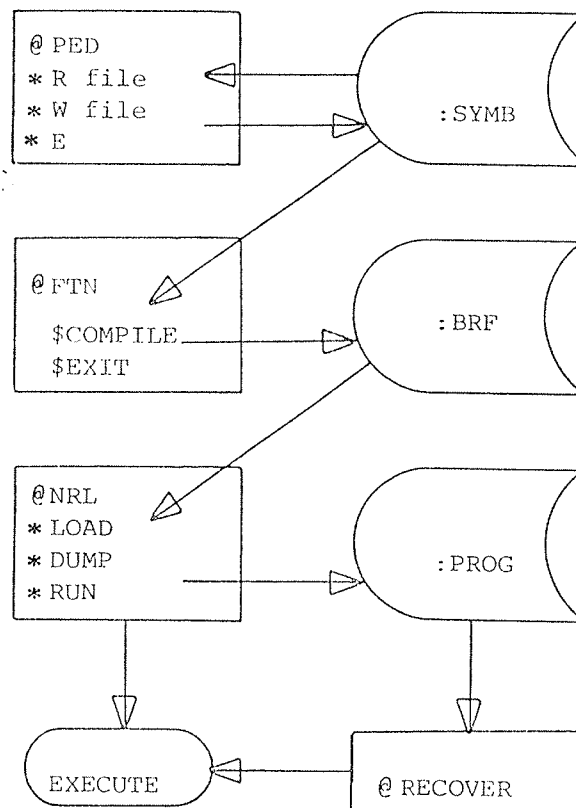


Fig. 10. File Types

File types are used to indicate the format of the information on the file. For example, most programs are created as SYMB files containing symbolic code. When it is compiled, normally a new file is created with the same <object name> but of type BRF (Binary Relocatable Format). A reloadable program has the type PROG. Subsystems and commands assume a default type when the file name is specified, without the type being specified.

3.2.4. File Versions

Files may also differ in the version number. If a file is created in more than one version, then no. 1 will be the latest version, no. 2 the previous one, and so on. When a file is opened for read, no. 1 will be accessed. When it is opened for write, the version with the highest no. will be selected. When writing is finished and the file is closed, its no. becomes 1 and all other versions are increased by one. It is also possible to access a specific version of a file by specifying its version number, e.g.:

@LIST-FILES OLE,,

will list all versions of file OLE, whereas:

@LIST-FILES OLE:1,,

will list only version no. 1.

3.2.5. Terminal Types

A large number of different terminals are used with the ND computers. They are either display terminals (VDU units) or hard copy terminals from different manufacturers. To make it possible to distinguish between the different types, SINTRAN III will store a terminal type number as part of the information about a LDN, provided it is a terminal. A standard terminal type name and corresponding number for each type is shown in appendix B of the REFERENCE MANUAL. If a terminal is sold as part of the system it is also listed with its ND-number (supplied by ND marketing).

If a user wants to get a particular terminal standardized, he should contact his ND representative to get it placed in the list. The terminal type is stored in the operating system and used by some of the SINTRAN III subsystems.

Two commands are used to manage the terminal type, @SET-TERMINAL-TYPE and @GET-TERMINAL-TYPE. The corresponding monitor calls are MSTTY (MON 17) and MGTTY (MON 16). They are explained in section 6.3.

3.2.6. Scratch Files

Each terminal (with a background program) has been assigned a scratch file. The file is indexed and is kept permanently open while the terminal user is logged in. When the user logs out the file is reduced to a maximum of 32 pages. LDN = 100 (octal) is permanently assigned to this file.

The scratch file can be used for temporary data on mass storage. Some subsystems use this file and it is then not available for the terminal user. However, the terminal user can specify 100 as BRF output from the FORTRAN compiler for example. The same number can then be specified as input to the ND Relocating Loader.

3.2.7. Logical Device Number

As explained in section 3.1, a file name must be associated with a number for data access. A file may be a mass storage file (OLE:SYMB;1) or a peripheral file (LINE-PRINTER;1). When opening the file from a command or a MAC program, a number is supplied by the file system on return. This number is called the logical device number (LDN). Note that this number is in general different from the FORTRAN unit number as used by a FORTRAN OPEN statement. The Unit no. will be described in section 3.2.8.

For example, consider the command:

```
@OPEN-FILE OLE,RW
FILE NUMBER IS 000101
@
```

We have opened a mass storage file, and the file system has returned an LDN = 101 (octal). A character device can be opened as follows:

```
@OPEN-FILE LINE-PRINTER,W
FILE NUMBER IS 5
@
```

LINE-PRINTER is a peripheral file and has 5 permanently assigned as its LDN.

The number 101 obtained above is an example of a dynamic LDN which is assigned on demand as a file is opened. A dynamic file number was also allocated for LINE-PRINTER above, but only the static LDN was used in the Open command. A static LDN is global for the whole system. A dynamic LDN, however, is local to the particular user. That is, if another user opens another file, he may also get the number 101.

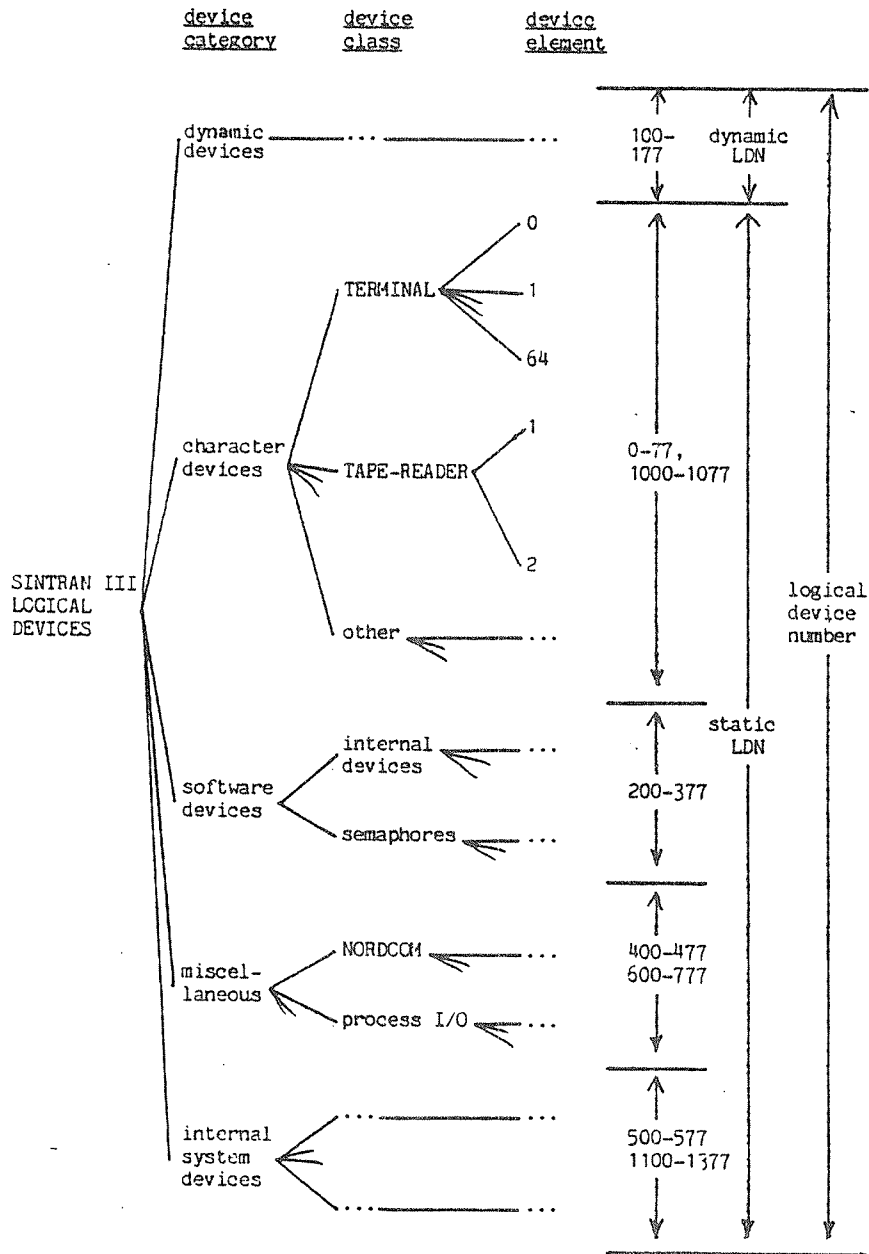


Fig. 11. Categories of Logical Devices

The structure of logical devices is shown in figure 11. The two most important main categories are dynamic devices, which are the opened files, and character devices. The way in which LDN's are assigned is shown at the right-hand side of the figure. (See appendix C of SINTRAN III REFERENCE MANUAL for a more detailed list.) Static file numbers are universal for all terminals but dynamic numbers are local to your own terminal. (An open file is only open for your own terminal.)

At the bottom of the hierarchy are the individual logical devices, each device being a device element. For the character device TERMINAL, this can be terminal 0, terminal 1, or terminal 2, etc. It is important to distinguish the element number from the LDN. For each device element of a character device (except terminal 0) there is also a peripheral file name, e.g., terminal no. 2 has peripheral file name TERMINAL, element no. 2, and LDN = 9. LDN for terminals are described separately in section 3.2.9. All devices attached to a SINTRAN system have a unique device number. An LDN can be reserved or released. Note that LDN numbers shown in figure 11 are in octal.

3.2.8. FORTRAN Unit Number

The unit number used in FORTRAN programs is similar to the LDN described in section 3.2.7. In an OPEN statement it is used for associating a value with the file name for later data access. As opposed to the LDN, a value must be supplied on input to the OPEN statement. This value is used throughout the rest of the data access. It is an advantage to use the same value as the LDN whenever possible.

Consider two cases of output from a program. Firstly, writing on the line printer. This can be done directly by a statement such as:

```
WRITE(5,10)I,J
```

5 is the standard LDN of the LINE-PRINTER. If 5 has been opened by the OPEN statement then 5 will be assumed to be a LDN.

Secondly, writing to a file. The file must first be opened with a unit number, e.g.:

```
IFIL=65  
OPEN(IFIL,FILE='TIME:DATA',ACCESS='W')
```

The file TIME:DATA will now be open for data access with the unit number 65 (=101 in octal). It can now be written to by a statement such as:

```
WRITE(IFIL,10)TD(7),TD(6),TD(5),TD(4),TD(3),TD(2),TD(1)
```

In general, the terminal user should avoid mixing LDN and unit numbers whenever possible.

3.2.9. Logical Device Numbers for Terminals

From a background program it is possible to write to or read from any terminal except the console terminal (TERMINAL 1). Mostly, we want to communicate with our own terminal and LDN = 0 and 1 are used for this purpose.

LDN = 0

This is only permitted for input. It specifies edited input from the terminal. Edited input means that the terminal user may use the control characters mentioned previously.

LDN = 1

This is permitted on both input and output. It specifies that all characters are input from or output to the terminal (except ESCAPE on input which terminates the program).

LDN = 9, 34, 35, etc.

This specifies input from or output to TERMINAL-2, TERMINAL-3, TERMINAL-4, etc. (Device element no. 2, 3, 4, etc.) Appendix C of SINTRAN III REFERENCE MANUAL lists the LDN's for all terminals.

3.3. File Directories

3.3.1. General

A directory is a set of files and has the structure as shown in figure 6. There can only be one directory per mass storage medium. Mass storage may be hard disks or floppy-disks. Each directory may contain files for one or several users. If the directory is contained on a removable medium, it may be moved to other installations and used there. Each mass storage medium to be used as a directory must be created with a name and entered before it can be used by the system. When a directory is no longer needed, it may be released and physically removed from the system. The next time it is needed, it must be physically replaced and entered again.

The first directory entered containing SINTRAN III (and related subsystems) is regarded as a main directory. This directory cannot be released.

Any directory in the system on a disk can be a default directory. A user may have space in n , $0 \leq n \leq$ total number of default directories. However, the directory with lowest index in which he has space when he enters will be default, and he must specify the directory name to access all the others. If a user leaves out the directory name in a file name, this default directory is assumed.

The system supervisor (user SYSTEM) is responsible for entering the main and default directories, setting them up as main or default, creating and allocating space to the different users. Time-sharing

users can enter and release all other directories. Time-sharing users can also create and administer floppy-disks.

3.3.2. Directory Commands

The following commands are used to establish and maintain directories. They are all generally available for time-sharing users, except when noted otherwise.

3.3.2.1. @CREATE-DIRECTORY

@CREATE-DIRECTORY <dir. name>,<device
name>[,<unit>],<bit-file address>

Establish a new directory on a mass storage medium. The only device permitted for time-sharing users is floppy-disk. <device name> must then be FLOPPY-DISK-1 or FLOPPY-DISK-2. <Unit> is only specified if more than one unit is connected to the same controller. (Note: If only one unit(No. 0)is installed, the unit number is neither necessary nor permitted in the command.) Usually the user chooses the default value of <bit-file address>. The location will then be chosen by the system. When allocating large contiguous files on a floppy-disk the bit-file should be allocated at the end of the disk. The value of the parameter would then be 150.

For example:

@CREATE-DIRECTORY FLOPPY-1,F-D-1,0,,

Create a new directory on the floppy disk unit 0 attached (connected) to controller with name FLOPPY-DISC-1. It is given the name FLOPPY-1. Section 3.11 shows how to initialise a floppy disk.

3.3.2.2. @ENTER-DIRECTORY

@ENTER-DIRECTORY <dir. name>,<device name>[,<unit>]

Make a directory available to the system. The parameters are the same as for @CREATE-DIRECTORY, except for bit file address, e.g.:

@ENTER-DIRECTORY F-1,F-D-1,0

This makes the directory, with the name FLOPPY-1, on FLOPPY-DISK-1 known to the system. It must have the name FLOPPY-1 or any superset of F-1 up to 16 characters will do.

3.3.2.3. @CREATE-USER

@CREATE-USER <dir. name>:<user name>

Enter a new user name on the specified directory. For time-sharing users the directory must be on a floppy-disk. The user name must already be known in a main directory, e.g.

@CREATE-USER F-1:GUEST

Create the new user named GUEST on directory F-1.

3.3.2.4. @GIVE-USER-SPACE

@GIVE-USER-SPACE <dir. name>:<user name>,<no. of pages>

Reserve the number of pages for the given user on the directory. For time-sharing users the directory must be on a floppy-disk. The maximum number of pages that can be reserved on a floppy-disk is 148, e.g.:

@GIVE-USER-SPACE F-1:GUEST,148

3.3.2.5. @RELEASE-DIRECTORY

@RELEASE-DIRECTORY <dir. name>

Make the specified directory unavailable for the system. It may now be physically removed from the system, e.g.:

@RELEASE-DIRECTORY F-1

It is important to remember this before removing a floppy-disk from the floppy-drive.

3.4. File Creation and Deletion

Files can be created either by implicit file creation or explicit file creation. An example of how a file may be implicitly created, using the editor QED is:

*W "AAA"

This command creates the file AAA:SYMB and writes the text-buffer to this file. Files created in this way are always indexed files. Files can be created implicitly both in subsystems, such as QED, and by commands. An example of an explicit creation of a file is:

3.4.1. @CREATE-FILE

@CREATE-FILE <file name>,<no. of pages>

Create a new file with the given name and number of pages.

If <no. of pages> is zero, an empty file will be created. The first time the file is written to it will be marked as an indexed file.

If the file is expanded with EXPAND FILE the file will be contiguous.(provided enough space is available)

If <no. of pages> is greater than zero, an empty contiguous file will be created (provided enough space is available) with the indicated <no. of pages>. If the file name contains a version number, it specifies the number of versions to be created.

E.g.:

@CREATE-FILE FILE-FOUR,10

Create the file FILE-FOUR:DATA (DATA=default type) as a contiguous file with 10 pages (20K bytes). (Note that quotes are not used here, although the file is new.)

3.4.2. @EXPAND-FILE

@EXPAND-FILE <file name>,<no. of pages>

This command must be used in order to expand a contiguous file. The parameter specifies the additional number of pages to be added to the file. For example:

@EXPAND-FILE FILE-FOUR,9

FILE-FOUR:DATA is expanded by nine pages (provided the contiguous space is available).

3.4.3. @CREATE-NEW-VERSION@CRFATE-NEW-VERSION <file name>,<no. of pages>

Create one or more new versions of the specified file and give them the number of pages as indicated. The rule for <no. of pages> is the same as for @CREATE-FILE. The number of new versions to be created depends on how many versions exist and the version number on the file name of the command. Assume that there are two versions of the file F-1:DATA.

F-1:DATA;1

F-1:DATA;2

If we give the command,

@CREATE-NEW-VERSION F-1;;4

This command will create new versions up to and including 4 (F-1:DATA;3 and F-1:DATA;4). Assuming instead that we had given the command:

@CREATE-NEW-VERSION F-1;;1,10

The new version created (containing no data) would be named as version 1. The previous version 1 would become version 2 and the old version 2 would become version 3. A new version of a file can only be created by this command.

3.4.4. @ALLOCATE-FILE@ALLOCATE-FILE <file name>,<page address>,<no. of pages>

In the above commands we have allowed the file system decide where to physically locate the file. It is possible to determine the location of a file by @ALLOCATE-FILE. This command is similar to @CREATE-FILE, but the file will be located at the <page address>. In order that the file should be created, the area in the directory must not be in use. It is also possible to allocate further versions of the file, in which case they will be located one after the other in the directory. For example:

@ALLOCATE-FILE FILE-THREE,100,8

An allocated file, FILE-THREE:DATA containing 8 pages, is created in default directory starting at page 100 (octal).

3.4.5. @ALLOCATE-NEW-VERSIONS

@ALLOCATE-NEW-VERSIONS <file name>,<page address>,<no. of
pages>

This command is similar to @CREATE-NEW-VERSION, but instead one or more new versions are created with the first version starting at <page address>. For example:,

@ALLOCATE-NEW-VERSIONS FILE-THREE;2,140,8

Version 2 of the contiguous file FILE-THREE:DATA is created starting at location 140. The file size is 8 pages.

3.4.6. @RENAME-FILE

@RENAME-FILE <old file name>,[<new object name> :<new type>]

Change object name and/or type of a file, e.g.:

@RENAME-FILE (PACK-TWO:GUEST)F-1:SRC,:SYMB

Change the type of the file from SRC to SYMB.

3.4.7. @DELETE-FILE

@DELETE-FILE <file name>[<:type>]

For deletion of a single file. The file type must be given. If a version number is given, only that version is deleted. If not, all versions will be deleted. The corresponding monitor call MDLFI (MON 54) is explained in section 3.4.10. E.g.:

@DELETE-FILE F-1:DATA

Deletes all versions of the file F-1:DATA.

3.4.8. @DELETE-USERS-FILES

@DELETE-USERS-FILES <file name>,<MANUAL CHECK?>

Delete all files matching <file name>. If the last parameter is YES each file name will be printed out. The user can then decide whether

the file should be deleted or not. If NO, all matching files will be deleted at once, e.g.:

@DELETE-USERS-FILES ,NO

Deletes all files in the users default directory.

3.4.9. @SET-TEMPORARY-FILE

@SET-TEMPORARY-FILE <file name>

Define the contents of a file as temporary. The pages of such a file will be deleted after it has been read from. The command should be given immediately after the file is created. This command is mostly used for spooling files, e.g.:

@SET-TEMPORARY-FILE LINE-PRINTER;2

Declare the file LINE-PRINTER, version 2 to be a temporary file.

Two commands related to file creation are declared as system commands. They are @SET-TERMINAL-FILE and @SET-PERIPHERAL-FILE, and are described in the manual SINTRAN III SYSTEM SUPERVISOR.

3.4.10. MDLFI (MON 54)

Delete a single file. The same rule applies to the file name as for @DELETE-FILE, e.g.:

LDX (FILE	% X REG POINTS TO FILE NAME STRING
MON 54	% MDLFI
MON 64	% ERMSG ON ERROR RETURN
...	% NORMAL RETURN

FILE, 'FILE-THREE:DATA'

This deletes the file FILE-THREE:DATA.

3.5. File Access Permission and Reservation

A file may be accessed for read, write, append (i.e., add data to the end of the file), common (i.e., more than one user having access at the same time), and directory (i.e., creation, deletion, change access).

The access that you are allowed to do to a file depends upon which user type you belong to and what kind of access the owner of the file has permitted for your category. More about this in the next section.

3.5.1. User Types

The users of a file are divided into three categories:

OWN: The owner of the file. The normal default access permitted for the owner of a file is all types of access.

FRIENDS: A group of users designated by a given user. (They are friends relative to a users files.) Normal default access types permitted are read, write, and append.

PUBLIC: All other users. The normal default access type is read.

You obtain the default access type associated with a file when the file is created (For example, @CREATE-FILE) The term "normal default" signifies the default when the system is started. It can be changed later on by the command @SET-DEFAULT-FILE-ACCESS (to be explained below).

The protection given to a file is illustrated in figure 12. It is normally organised such that the file owner has the greatest access while the public has the least.

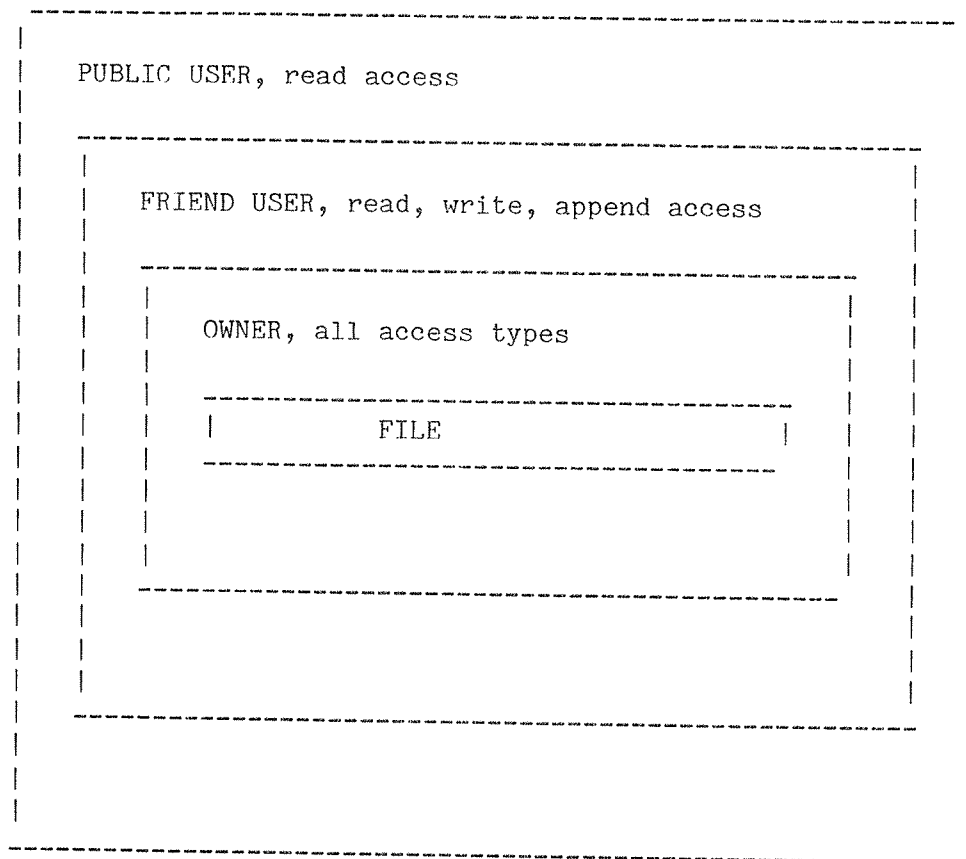


Fig. 12. Normal File Protection

3.5.2. Granting Access to Files

The file access-type is stored in the directory as part of the information about each file. In addition, for each user there is a special list of access-types for friends (see figure 13). For the users own and public requests, only the permission stored with the file is checked. For friend access, the permission for the friend is checked in addition to the permission stored with the file. The access is granted only if it is permitted in both lists.

For commands that set the permitted access for individual files we have:

3.5.3. @SET-FILE-ACCESS

@SET-FILE-ACCESS <file name>, <public access>, <friend access>, <own access>

Which sets the access mode for a specific file. The access specification is any relevant combination of:

@CREATE-FRIEND GUEST

Declare the user GUEST as your friend.

3.5.6. @DELETE-FRIEND@DELETE-FRIEND <user name>

Remove the user name from the list of your friends. He will now be a public user, e.g.:

@DELETE-FRIEND GUEST

The user GUEST is removed from the list of your friends.

	permission stored with friend information		permission stored with the file	
own			own permission (@SET-FILE- ACCESS)	granted
request				rejected
friend	friend permission (@SET- FRIEND- ACCESS)	granted	friend permission (@SET-FILE- ACCESS)	granted
request		reject- ed		rejected
public			public permission (@SET-FILE- ACCESS)	granted
request				rejected

Fig. 13. Granting Access to Files.

3.5.7. @SET-FRIEND-ACCESS

@SET-FRIEND-ACCESS <user name>,<access type>

Specify access from a friend to the users own files. Figure 13 shows how this access permission is used together with the file permission.

Friend access for a file is WA. By typing:

@SET-FRIEND-ACCESS GUEST,RW

Access to the file for user GUEST is now only W (It may only be opened with write access). E.g.:

File access for a friend(SET-FILE-ACCESS):

```
R|W|A|C|D|
-|-|-|-|-|
```

@FILE-STATISTICS

```
FILE 0 : (PACK-ONE:USER1)MYFILE:SYMB;1
          (INDEXED FILE)
          PUBLIC ACCESS : READ
          FRIEND ACCESS : WRITE, APPEND
          OWN ACCESS : READ, WRITE, APPEND, COMMON, DIRECTORY
          OPENED 7 TIMES
          CREATED 10.10.43  AUGUST 16, 1982
          OPENED FOR WRITE 10.10.43  AUGUST 16, 1982
          1 PAGE , 65 BYTES IN FILE
```

Access for a friend(SET-FRIEND-ACCESS):

```
R|W|A|C|D|
-|-|-|-|-|
```

@LIST-FRIENDS

```
FRIEND 0 : GUEST
          ACCESS : READ, WRITE
```

Friends access to the file(The intersection of the two):

```
R|W|A|C|D|
-|-|-|-|-|
```

3.6. Requesting Access to Files

Files are accessed both from commands and monitor calls. The access is only accepted if the corresponding file (and friend) access allows it, as shown in the previous section. There are two ways of requesting access, explicitly (commands and monitor calls) or implicitly (commands only).

With explicit access mode you specify which type of access you want by a character combination. The characters are:

- R - read request
- W - write request
- X - random access request
- A - append request
- C - common request (only for contiguous files)
- D - direct transfer (only for foreground programs)

For legal combinations of these see SINTRAN III REFERENCE MANUAL (ND-60.128).

An example is @OPEN-FILE, the FORTRAN OPEN statement or OPEN (MON 50).

@OPEN-FILE OLE,RX

means that the file is opened for random read only. In FORTRAN a similar statement would be:

```
OPEN(IFIL,FILE='OLE',ACCESS='RX')
```

The file OLE is opened for random write only.

With implicit access mode the access mode is implied by the command:

@COPY-FILE (SYSTEM)OLE,(SYSTEM)PER

means that you are requesting read access to (SYSTEM)PER:SYMB and write access to (SYSTEM)OLE:SYMB.

3.7. Reserving and Releasing Files

The previous sections showed that there is a mode of access called common when the file is opened. It is also possible to reserve the access to a file for the users terminal. (Note that it is the terminal and not the user name.) Using commands it is possible to reserve peripheral files by name and whole mass storage devices, while with monitor calls it is possible to reserve all static LDNs (see figure 11).

It is not possible to reserve mass storage files and dynamic LDNs.

3.7.1. @RESERVE-FILE

@RESERVE-FILE <peripheral file name>

Reserve a peripheral for the exclusive use of the users terminal. (The corresponding monitor call RESRV (MON 122) is described in section 3.7.5.) For example:

@RESERVE-FILE LINE-PRINTER

The peripheral file LINE-PRINTER is reserved for the exclusive use of the users terminal.

3.7.2. @RELEASE-FILE

@RELEASE-FILE <peripheral file name>

Permit a peripheral to be used from other terminals.

3.7.3. @RESERVE-DEVICE-UNIT

@RESERVE-DEVICE-UNIT <device name>[,<unit>,<'F' or 'R'>,<subunit>]

Reserve a device for special use, such that the directory on the device can not be entered. For time-sharing users this should be restricted to FLOPPY-DISK. For example:

@RESERVE-DEVICE-UNIT F-D-1 0

Do not allow FLOPPY-DISK-1, unit 0 to be used from other terminals.

3.7.4. @RELEASE-DEVICE-UNIT

@RELEASE-DEVICE-UNIT <device name>[,<unit>,<'F' or 'R'>,<subunit>]

This is the release command corresponding to the reserve command above.

The monitor calls for reserving and releasing logical devices are RESRV and RELES.

3.7.5. RESRV (MON 122)

```
INTEGER RESRV  
ISTAT=RESRV(<LDN>,<read/write>,<return flag>)
```

Reserve a static LDN for the exclusive use of this program. If <read/write> is zero, then the input part is to be reserved, if one, the output part is reserved. <return flag> specifies the return strategy to be used if the LDN is already reserved. If zero, the program is to be set in a waiting state. If one, the call will return with a function value of -1. Successful reservation always gives a function value of zero.

If the user program exits without releasing the LDN, then it will not be released until the user logs out. (User RT and SYSTEM can always force it to be released by the command @PRLS.)

For example:

```
INTEGER RESRV  
ISTAT=RESRV(5,1,0)
```

Reserve the output part of LDN = 5 (usually LINE-PRINTER). If the device is already reserved the program will be set in a waiting state. It will be started again when the device is released.

3.7.6. RELES (MON 123)

```
CALL RELES(<LDN>,<read/write>)
```

This is the release routine corresponding to RESRV. A time-sharing user can only release a LDN via the terminal from which it was originally reserved.

For RT-programming there are two additional commands for reserving and releasing, @PRSRV and @PRLS. They will force the reserving or releasing independently of its current state.

In RT-programs the monitor call WHDEV (MON 140) can be used to find out who has reserved a LDN.

3.8. Reading and Writing Data

3.8.1. General

This section describes commands and monitor calls for transmitting data between files and the user's memory.

Most of the functions are performed by monitor calls since reading and writing data usually takes place in programs. Some of the calls have corresponding commands (e.g., OPEN and @OPEN-FILE).

3.8.2. Opening Files

The most frequently occurring command and monitor call is @OPEN-FILE and OPEN (MON 50). This function use LDN (file number). In the FORTRAN OPEN statement we use the unit number as described in section 3.2.8. The other functions are commands to open a file with a specific LDN, to open it as a scratch file, and to set a file permanently open.

RT-users have two special commands for opening a file for RT-programs. They are @RTOPEN-FILE and @RTCONNECT-FILE.

3.8.2.1. @OPEN-FILE

@OPEN-FILE <file name>,<access type>

Make a file available for data access and return the LDN to be used for accessing the file. The <access type> requested is checked with the legal access types for the file, as described in sec. 3.5.3. Only certain combinations are legal. The file can be created with the @OPEN-FILE command. The corresponding monitor call is described in section 3.8.2.5, e.g.:

```
@OPEN-FILE "PERSONNEL-1",WA
FILE NUMBER IS 000101
@
```

Create and open the file PERSONNEL-1:SYMB for write and append access. The file is given the number 101 (octal).

```
@OPEN-FILE (SYSTEM)OLE:SYMB,RW
```

```
NOT READ AND WRITE ACCESS
@
```

The command requests the file OLE:SYMB to be opened for read and write access. This example shows an unsuccessful result, and the reason for this is that the access was not set to allow you to open the file for

read and write.

3.8.2.2. @CONNECT-FILE

@CONNECT-FILE <file name>,<file no.>,<access type>

Open a file with the specified dynamic LDN. The call is similar to @OPEN-FILE except that the user specifies the file no, e.g.:

@CONNECT-FILE OLE,110,RW
@

Open OLE:SYMB for sequential read and write access. Its file no. will be 110 (octal).

3.8.2.3. @SCRATCH-OPEN

@SCRATCH-OPEN <file name>,<access type>

Open the file as a scratch file. Such a file is only closed when logging out, or by the command @CLOSE-FILE -1 which closes all files not set permanent open. When closed, no more than 32 of the file pages are retained. In other respects, the command is similar to @OPEN-FILE, e.g.:

@SCRATCH-OPEN TEMP,RWA
FILE NUMBER IS 000104
@

Open the file TEMP:SYMB as a scratch file for read, write, and append access. The file number 104 is allocated to the file.

3.8.2.4. @SET-PERMANENT-OPENED

@SET-PERMANENT-OPENED <LDN>

The already opened file cannot be closed by the command @CLOSE-FILE -1. For example

@OPEN-FILE OLE,RW
FILE NUMBER IS 101
@SET-PERMANENT-OPEN 101

...

@CLOSE -1

the file is still open. The command @CLOSE -1 does not close files opened by the command @SET-PERMANENT-OPENED.

The command @LIST-OPEN-FILES, will list open files.

...

@CLOSE 101

the file 101 is now closed

3.8.2.5. OPEN (MON 50)

Open a file for data access. The file name can be read either from the user program or the user terminal as follows:

LDX (FILE	% X = addr. of name
LDA (FILTY	% A = addr. of file type
SAT 4	% T = sequential read
	% and write
MON 50	% OPEN
MON 64	% error return
...	% normal return
FILE, 'PER'	% file name string
FILTY, 'SYMB'	% file type string

For other access codes see the SINTRAN III REFERENCE MANUAL.

Open the file PER:SYMB for sequential read and write.

3.8.3. Sequential Input/Output

The following are monitor calls available from user programs. Selecting a call depends on how many characters you want to be read by one call, what type of return strategy you want etc. There is a corresponding output call for each input call (though not vice versa).

The 8-byte calls (M8INB, M8OUT, B8INB, B4INW and B8OUT) may only be used for character devices and software devices (figure 11) and NORD-NET. These calls are described in detail in the SINTRAN III REFERENCE MANUAL.

3.8.3.1. INBT (MON 1, FORTRAN call: INCH)

Read one byte from a device. (Read one word if the device is a data-link or word-oriented device.) For example

```
ICH=INCH(2)
IF(ERRCODE.NE.0)GO TO 1000
```

Read one character to ICH from Fortran unit number 2. The read was successful if ERRCODE is zero, if not it will contain the error number.

3.8.3.2. OUTBT (MON 2, FORTRAN call: OUTCH)

This is the output call corresponding to INBT above, e.g.:

```
CALL OUTCH(5,ICHAR)
IF(ERRCODE.NE.0)GO TO 1000
```

Output one character in ICHAR to Fortran unit number 5. ERRCODE contains error status on return.

3.8.3.3. INSTR (MON 161)

Read a string of characters from a peripheral device. (This call is an optional part of SINTRAN III and can be requested when ordering SINTRAN III from ND.) For example:

```
DIMENSION MTXT(50)
...
ISTAT=INSTR(1,MTXT,100,15B)
IF(ISTAT.NE.0)GO TO 1000
```

Read a string of characters from your own terminal to MTXT. Terminate after 100 characters are read or after receiving CARRIAGE RETURN (15B), whichever occurs first.

3.8.3.4. OUTST (MON 162)

Write a string of characters to a peripheral device. (This call is optional as in 3.8.3.3 above.) For example:

```
INTEGER OUTST
DIMENSION MOUT(50)
...
ISTAT=OUTST(1,MOUT,15B)
```

Write a string of characters from MOUT to your terminal. Terminate after the first CARRIAGE RETURN (15B).

3.8.3.5. MSG (MON 32)

Write a character string on your terminal, e.g.:

```
LDX (TEXT          % X = addr. of character
MON 32             % string
...               % MSG
...               % normal return
TEXT, 'ERROR IN INPUT DATA' %the text is terminated with '
```

The text ERROR IN INPUT DATA is written on your terminal.

3.8.3.6. IOUT (MON 35)

Print a number in octal or decimal format on your terminal, e.g.:

SAT	12	% print in decimal format
LDA	(NUMB	% A = number
MON	35	% IOUT
...		% normal return

The value of NUMB is written in decimal format.
SAT 22 instead of SAT 12 will print in octal.

3.8.3.7. NOWT (MON 36)

Set wait mode or no wait mode of I/O calls to character devices.

NOWT (no-wait) modifies the behaviour of INBT (INCH) and OUTBT (OUTCH) in the event of the device not being ready.

On input, the normal situation is that INBT will wait for the next character to be input, if one is not immediately available. However, NOWT may be used to cause INBT to return immediately if there is no character immediately available and, in this case, give an error code of 3. (End of file.)

On output, the normal behaviour of OUTBT if the device cannot receive the character, is to wait. With NOWT one can modify this to allow OUTBT to return immediately. In the example, below HOLD is used to wait for I/O completion.

```
DIMENSION MTXT1(1000),MTXT2(1000)
...
C INITIATE OUTPUT TO LINE PRINTER 1 AND 2
  ISTAT=NOWT(5,1,1)
  IF(ISTAT.NE.0)GO TO 999
  ISTAT=NOWT(13,1,1)
  IF(ISTAT.NE.0)GO TO 999
  I=1
  J=1
C
C OUTPUT ARRAY MTXT1 AND MTXT2
100  IF(I.GT.1000)GO TO 200
C
C      OUTPUT TO LINE PRINTER 1
      CALL OUTCH(5,MTXT1(I))
      IF(ERRCODE.EQ.0)I=I+1
      IF(ERRCODE.NE.0.OR.ERRCODE.NE.3)GO TO 999
C
C      OUTPUT TO LINE PRINTER 2
200  IF(J.GT.1000)GO TO 300
      CALL OUTCH(13,MTXT2(I))
      IF(ERRCODE.EQ.0)J=J+1
      IF(ERRCODE.NE.0.OR.ERRCODE.NE.3)GO TO 999
...

```

```

300      CALL HOLD(1000,4)
        GO TO 100
C
C CONTINUE IF ANY PRINTER HAS MORE DATA
      STOP
C
C ERROR EXIT
999      ...
        STOP
        END

```

The program will output the array MTXT1 containing 1000 characters to line printer 1 and the array MTXT2, also containing 1000 characters, to line printer 2. The first statements set the peripherals in no-wait-mode and initiate indexes to the arrays. The program loop consists of two sets of output statements. The call to OUTCH attempts to put a character in the output buffer. If successful, the index can be incremented. If ERRCODE is 3 the index will not be incremented and an attempt will be made to output the same character at the next call to OUTCH for the same peripheral. If ERRCODE is neither 0 nor 3 there is an error and the exit is taken through statement 999.

HOLD operates in a special way in NO WAIT-mode. If a break condition is detected (i.e., a peripheral has signaled that a break character has been transmitted), there will be an exit from the call. The call to HOLD above then behaves as a "wait for break" statement. (The maximum hold time is specified in the call.) When any of the peripherals generate a break, HOLD will exit and the program will attempt to output the next character or the same character again. The output will continue until there is no further data for any device.

3.8.4. Random Input/Output

Random I/O can be used to access both indexed and contiguous files. A random file consists of a set of equal sized records. The records are addressed in terms of blocks. The standard block size is 256 16-bit words. The size can be modified by @SET-BLOCK-SIZE or the parameter RECL (record length) in the FORTRAN OPEN statement.

The two most important monitor calls for this type of I/O are RFILE and WFILE. They are usually used as monitor calls, but are also available as commands. One call transmits a number of words which is independent of the block size.

3.8.4.1. @RFILE

@RFILE <file no.>,<memory address>,<block no.>,<no.
of words>

Transfer <no. of words> from one or more random file blocks into the user's memory where <no. of words> is the number of words to be transferred. The number is independent of the block size, therefore, one or more blocks may be read, e.g.:

@RFILE 101,400,0,1000

(All parameters are in octal.) Read 1000B words from block 0 of file 101 to memory address 400. The block size is 400B (standard), so two blocks are read. Also, a <no. of words> less than the block size may be specified.

3.8.4.2. @WFILE

@WFILE <file no.>,<memory address>,<block no.>,<no.
of words>

Transfer <no. of words> from the user's memory to one or more blocks in a random file. For example:

@WFILE 101,400,0,1000

(All parameters are in octal.) Write 1000 words from memory address 400 to block 0 of file 101. The block size is 400 (standard), so two blocks are written.

These two commands also exist as monitor calls as follows:

3.8.4.3. RFILE (MON 117)

Read a block from a file to the user's memory. For example:

```
DIMENSION MDATA(512)
...
CALL RFILE(101B,0,MDATA,0,512)
```

Read a block from file 101 to array MDATA. The extent starts in block 0 and is 512 decimal words long. The second parameter specifies that there will be no return from the call until all input is complete. It is the only permissible value for time-sharing users.

3.8.4.4. WFILE (MON 120)

Write an extent to a file from the user memory, e.g.:

```
DIMENSION MDATA(512)
...
CALL WFILE(101B,0,MDATA,0,512)
```

Write an extent to file 101 from array MDATA. The extent starts in block 0 and is 512 words long. The second parameter must be zero for time-sharing users.

Four other monitor calls can be used for random file access. They are RDISK (MON 5), WDISK (MON 6), RPAGE (MON 7), and WPAGE (MON 10). These calls are only included in SIII in order to be compatible with the old TSS operating system and they should preferably be avoided by "new" users.

3.8.5. Closing Files

Files can be closed both from commands and monitor calls. If a file is open when the program terminates, it will be closed automatically, unless permanent open. This applies to both normal and abnormal termination. (The user can avoid this by setting the file permanently open, see next section.)

If the file is opened by a command, it will always be closed when the user logs out.

3.8.5.1. @CLOSE-FILE

@CLOSE-FILE <LDN>

Close one or more files. If the file number, <LDN>, = -1 then all files not permanently opened (see section 3.8.2.4) are closed. If = -2 then all files are closed. For example:

@CLOSE 101

Close the file number 101.

3.8.5.2. CLOSE (MON 43)

The monitor call can close files in a manner similar to the command. For example:

SAT	-1	% close all files not permanently opened
MON	43	% CLOSE
JMP	ERROR	% error return
...		% normal return

This call has a function similar to the command @CLOSE-FILE -1.

3.8.6. Data Access Parameters

This section deals with the parameters used for sequential and random data access of a file. For sequential access there is a byte pointer which points to the next byte to be accessed. The maximum byte pointer is one less than the number of bytes in the file. The block size may be modified and the byte pointer may be set to the beginning of a block.

The block size is the only parameter relevant for random access.

ECHOM (MON 3) and BRKM (MON 4) affect data access for terminals. The calls are documented in chapter 6.

3.8.6.1. @SET-BYTE-POINTER

@SET-BYTE-POINTER <file no.>,<byte no.>

Set the byte pointer to be used by the next sequential file access. The corresponding monitor calls are SETBT and REABT, e.g.:

@SET-BYTE-POINTER 101,0

The byte pointer is reset to the beginning of the file.

3.8.6.2. @SET-BLOCK-SIZE

@SET-BLOCK-SIZE <file no.>,<block size>

Set the block size to be used by the next file access. The standard block size is 256 decimal words. The corresponding monitor call is SETBS. For example:

@SET-BLOCK-SIZE 101,512

Set the block size to 512 decimal words.

3.8.6.3. @SET-BLOCK-POINTER

@SET-BLOCK-POINTER <file no.>,<block no.>

Set the byte pointer to the beginning of a block. The position is then dependent on the block size. The corresponding monitor call is SETBL. For example:

@SET-BLOCK-SIZE 101,512
@SET-BLOCK-POINTER 101,1

Set the byte pointer to byte 1024 in the file. (The first byte has address 0.)

3.8.6.4. SETBT (MON 74)

Set the byte pointer of a file, e.g.:

```
DOUBLE INTEGER IBYTE
...
IBYTE=0
CALL SETBT(10,IBYTE)
```

Reset the byte pointer of LDN = 10 to 0.

3.8.6.5. REABT (MON 75)

Read the byte pointer of a file, e.g.:

```
DOUBLE INTEGER IBYTE
...
CALL REABT(10,IBYTE)
```

Read the byte pointer of LDN=10.

3.8.6.6. SMAX (MON 73)

Set the maximum byte pointer of a file, e.g.:

```
DOUBLE INTEGER MAXBY
...
MAXBY=377777B
CALL SMAX(10,MAXBY)
```

Set the maximum byte pointer of LDN=10 to 256K-1. The maximum size of the file will then be 256K bytes.

3.8.6.7. RMAX (MON 62)

Read the maximum byte pointer of a file, e.g.:

```
DOUBLE INTEGER MAXBY
...
CALL RMAX(10,MAXBY)
```

Read the maximum byte pointer of LDN = 10.

3.8.6.8. SETBS (MON 76)

Set the block size of an open file. The standard block size is 256 decimal words, e.g.:

CALL SETBS(101b,512)

Set the block size of LDN = 101 (octal) to 512 (decimal) words. The block size can also be set by including RECL= ... in the OPEN statement.

3.8.6.9. SETBL (MON 77)

Set the byte pointer to the beginning of a block. The position in the file is then dependent on the block size, e.g.:

CALL SETBS(101B,512)
CALL SETBL(101B,1)

Set the byte pointer to byte 1024 (decimal) in the file. (The first byte has address 0.)

3.8.6.10. CIBUF (MON 13)

Clear the input buffer of a device, e.g.:

CALL CIBUF(9)

Clear input buffer of terminal 2. (Logical Device Number 9.)

3.8.6.11. COBUF (MON 14)

Clear the output buffer of a device, e.g.:

CALL COBUF(9)

Clear the output buffer of terminal 2. (Logical Device Number 9.)

3.8.6.12. ISIZE (MON 66)

Get the current number of bytes in an input buffer, e.g.:

NUMB=ISIZE(9)

On return NUMB will contain the number of bytes in the input buffer of terminal 2.

3.8.6.13. OSIZE (MON 67)

Get the current number of bytes in an output buffer, e.g.:

INTEGER OSIZE

...
NUMB=OSIZE(9)

On return NUMB will contain the number of bytes in the output buffer of terminal 2.

3.9. Commands for Copying Files

Data can be copied from a source file to a destination file. The methods of copying are: one character at a time or one page at a time. It is also possible to copy more than one file by using the @COPY-USERS-FILES command in the BACKUP-SYSTEM.

User SYSTEM has available commands for copying all files in a directory (@COPY-DIRECTORY) or copying all pages on a device (@COPY-DEVICE).

3.9.1. @COPY

@COPY <destination file>,<source file>

Copy the contents of the source file one byte at a time to the destination file, e.g.:

@COPY OLE,TAPE-READER

copy from the tape-reader to OLE:SYMB.

3.9.2. @COPY-FILE

@COPY-FILE <destination file>,<source file>

Copy the contents of the source file one page at a time to the destination file. This command should be used if both files are mass storage files. For example:

@COPY-FILE OLE,PER

Copy the contents of PER:SYMB to OLE:SYMB.

You should note that @COPY-FILE is much faster than @COPY.

3.10. Commands for Listing File System Information

Commands are available for listing information about files, directories, users and friends. Some commands are similar to others, except that they give more details. (@FILE-STATISTICS give more details than @LIST-FILES.) The command @LIST-OPENED-FILES gives information about the files open by the own user. However, user RT and

SYSTEM have available a similar command for files opened by RT-programs. (@LIST-RTOPENED-FILES).

Some monitor calls also give information about the file system. Reading the byte pointer and its maximum value (REABT and RMAX) was shown in section 3.8.6. For RT-programs a monitor call is available for checking whether a device is reserved or not (WHDEV, see section 3.5).

3.10.1. @LIST-FILES

@LIST-FILES <file name>,<output file>

List names of files matching <file name> on <output file>. The number listed (FILE ...) is the file object number, e.g.:

@LIST-FILES PROG,TERM

FILE 0 : (PACK-ONE:GUEST)PROG-1:SYMB;1
FILE 12 : (PACK-ONE:GUEST)PROG-2:SYMB;1
FILE 13 : (PACK-ONE:GUEST)PROG-2:BRF;1
@

The files matching PROG are listed on TERMINAL.

3.10.2. @FILE-STATISTICS

@FILE-STATISTICS <file name>,<output file>

List complete information about files matching <file name> on <output file>. For example:

@FILE-STATISTICS PROG-2:SYMB,TERM

FILE 12 : (PACK-ONE:GUEST)PROG-2:SYMB;1
 (INDEXED FILE)
 PUBLIC ACCESS : READ
 FRIEND ACCESS : READ, WRITE, APPEND
 OWN ACCESS : READ, WRITE, APPEND, COMMON, DIRECTORY
 OPENED 79 TIMES
 CREATED 16.10.32 APRIL 23, 1979
 OPENED FOR READ 21.59.18 AUGUST 15, 1979
 OPENED FOR WRITE 10.45.15 APRIL 25, 1979
 1 PAGE, 1000 BYTES IN FILE
@

3.10.3. @LIST-OPENED-FILES@LIST-OPENED-FILES <output file>

List dynamic file number and full name of all files opened for the terminal user. Note that this is the dynamic and not the static number, e.g.:

```
@LIST-OPENED-FILES,,
FILE NUMBER 000100 : (BIG-PACK:SCRATCH)SCRATCH08:DATA;1
@
Opened files are listed( on TERMINAL in this example).
```

3.10.4. @LIST-DIRECTORIES-ENTERED@LIST-DIRECTORIES-ENTERED <directory name>,<output file>

List names of entered directories matching <directory name> and devices where the directories are mounted, e.g.:

```
@LIST-DIRECTORIES-ENTERED,,,
DISC-2-75MB-1 UNIT 0 SUBUNIT 1: PACK-TWO
DISC-2-75MB-1 UNIT 0 SUBUNIT 0: PACK-ONE
@
All directories are listed (on the TERMINAL in this example).
```

3.10.5. @DIRECTORY-STATISTICS@DIRECTORY-STATISTICS <directory name>,<output file>

List information about directories matching <directory name> on <output file>, e.g.:

```
@DIRECTORY-STATISTICS PACK-TWO,,
DISC-2-75MB-1 UNIT 0 SUBUNIT 1 : PACK-TWO
(DEFAULT DIRECTORY)
1783 PAGES UNRESERVED AND 8847 PAGES UNUSED OUT OF 36945 PAGES
@
```

3.10.6. @LIST-USERS

@LIST-USERS [<directory name>:]<user name>,<output file>

List names of all users matching [<directory name>:]<user name>, e.g.:

```
@LIST-USERS PACK-TWO:,,  
USER 0 : PACK-TWO:SYSTEM  
USER 1 : PACK-TWO:JFB  
USER 2 : PACK-TWO:GUEST  
@
```

3.10.7. @USER-STATISTICS

@USER-STATISTICS [<directory name>:]<user name>,<output file>

List complete information about all users matching [<directory name>:]<user name> on <output device>. The default access used when creating a file for the user is listed if the directory is a main directory, e.g.:

```
@USER-STATISTICS PACK-ONE:SYS,,  
    CREATED 13.49.27 JANUARY 2, 1979  
    LAST DATE ENTERED 12.39.49 NOVEMBER 9, 1981  
    DEFAULT PUBLIC ACCESS : READ  
    DEFAULT FRIEND ACCESS : READ, WRITE, APPEND  
    DEFAULT OWN ACCESS : READ, WRITE, APPEND, COMMON, DIRECTORY  
    10801 PAGES USED OUT OF 12793 PAGES  
@
```

3.10.8. @LIST-FRIENDS

@LIST-FRIENDS <user name>,<output file>

List names of friends to the user matching <user name>. The list is produced on the <output file>, e.g.:

```
@LIST-FRIENDS,,,  
FRIEND 0 : SYSTEM  
            ACCESS : READ, WRITE, APPEND  
FRIEND 1 : GUEST  
            ACCESS : READ, WRITE, APPEND  
@
```

All friends are listed together with their file access.

3.10.9. @LIST-VOLUME@LIST-VOLUME <device name>,<unit>,<output file>

List the identification of a file, i.e., the reel label and the file labels.

3.10.10. @WHERE-IS-FILE@WHERE-IS-FILE <file name>

Check whether a file is opened and/or reserved. Also, list the user or RT-program who has opened or reserved the file, e.g.:

```
@WHERE-IS-FILE (SCR)SCRATCH08:D
SCRATCH08:DATA : OPENED BY USER GUEST ON TERMINAL 39
@
```

3.10.11. @LIST-DEVICE@LIST-DEVICE <LDN>,<read/write>

List name of the RT program having reserved the LDN. <read/write> = 0 for read part, 1 for write part, e.g.:

```
@LIST-DEVICE 1,1
RESERVED BY PROGRAM : BAK01
@
```

The output part of device 1 is reserved by the program BAK01.

3.11. Initialising a Floppy-Disk

Most time-sharing users will use floppy-disks for backup of programs and data. A floppy-disk is normally used with a directory in the same way as hard disks. Before files can be written to the disk, a directory must be created. The procedure is as follows,

1. Turn on power on the floppy-disk drive.(Not usual on ND-100).
2. If it is a "virgin" floppy-disk it must first be formatted by the command

@DEVICE-FUNCTION <peripheral file name>,FORMAT-FLOPPY

For example:

@DEVICE-FUNCTION F-2-0,FORM-FLOP

The command takes 1-2 minutes to complete. If there are any unusable pages on the diskette, they will be marked by @DEVICE-FUNCTION and listed during the formatting. The diskette can still be used.

Note that the first parameter is <peripheral file name> as opposed to <device name>. More information on @DEVICE-FUNCTION is given in the next section (3.12).

Note: Diskettes from Norsk Data are formatted prior to delivery. This is not necessarily true of subsidiaries.

3. The directory can now be created by the command:

@CREATE-DIRECTORY <dir. name>,<device name>[,<unit>],
<bit-file address>

For example:

@CREATE-DIRECTORY BACKUP,F-D-2,0,,

4. Entering the directory is performed by the command:

@ENTER-DIRECTORY <dir. name>,<device name>,<unit>

For example:

@ENTER-DIRECTORY ,F-D-2,0

5. Establishing the user is carried out by the two commands:

@CREATE-USER <dir. name>:<user name>
@GIVE-USER-SPACE <dir. name>:<user name>,<no. of pages>

The total <no. of pages> available on a floppy-disk is 148. If there are bad pages, the no. of pages must be reduced by the number indicated in step 2 above.

For example:

@CREATE-USER BACKUP:GUEST
@GIVE-USER-SPACE BACKUP:GUEST,148

The file space for BACKUP:GUEST is now available. Before the diskette can be removed the @RELEASE-DIRECTORY command must be issued.

When using the floppy-disk the next time only steps 1 and 4 are necessary.

3.12. Manipulating Magnetic Tapes, Floppy Disks, Versatec and cassettes

A command used for device-dependent functions on magnetic tapes, floppy disks, Versatec and cassettes etc., is:

@DEVICE-FUNCTION <peripheral file name>,<function name>[,<optional parameter 1>,<optional parameter 2>]

Documentation on all of the possible function names will be found in the SINTRAN III REFERENCE MANUAL. However, some of the more frequently-used ones are:

1. Format floppy (see example in previous section, 3.11).

New address information is written on the diskette. The default format is 256 words per sector, 8 sectors per track.

2. Advance to EOF (Note that the tape is positioned immediately after the EOF).

This enables a user to skip over a file on a tape or on a floppy disc. For example:

@DEV-FU MAG-TAPE-1,ADV-TO-EOF,2

In order to advance through two end-of-file marks on magnetic tape unit 1.

3. Rewind (i.e., Rewind the tape to its start position).
4. Unload (i.e., unload the tape from the second reel to rewind it).
5. Select Parity and Density

This may be necessary in order to read a tape. For the selections available see the SINTRAN III REFERENCE MANUAL. The default value is zero.

The monitor call corresponding to @DEVICE-FUNCTION is MAGTP (MON 144) as shown below.

MAGTP (MON 144)

Select device function. It corresponds to @DEVICE-FUNCTION described above. For example, to format a floppy disc the call format is:

ISTAT = MAGTP(<function code>,<d>,<LDN>,<input format>,<d>)

where <d> indicates a dummy parameter.

The logical device number will be found in appendix C of the SINTRAN III REFERENCE MANUAL, and for floppy disk 1, unit 0, it is 1000 (octal). The function code is 41 (octal). (Documentation on call formats, codes and a full list of functions also appear in the SINTRAN III REFERENCE MANUAL under MAGTP (MON 144). The call then becomes:

```
      IDUM=0
      ISTAT=MAGTP(41B,IDUM,1000B,IDUM,IDUM)
      IF(ISTAT.GT.0)GO TO 100
      ...
      ok
      ...
100   error
      ...
```

3.13. Taking Backups

No matter how reliable a computer is, it will go wrong at some point. It may also happen that the user overwrites a wrong file, so the contents of that file is lost. In order to protect ones files from such mishaps, it is advisable to take a back-up periodically.

A disk-to-disk copy is usually done once a week by the computer installation staff. So, in the worst case, you might lose a week's work. However, you can take your own back-up, of all or part of your files, when you have something that you particularly do not want to lose.

Back-ups can be taken on a large disk, magnetic tape, or floppy.

A special program, called BACKUP-SYSTEM, is available for taking backup. This program has several functions, but most users will use it mainly for copying files. The COPY-USERS-FILES command can be used both for taking the back-up and for recovering, should you have to. Two sets of parameters are required, one to describe what you are copying to, the other to describe the source, i.e. , what you are copying from. The files you have been working on are in a directory, on disk. The back-up files will also be on a directory if they are on a large disk or floppy. On tape they will be on a volume.

The backup is taken by starting the BACKUP-SYSTEM program and giving the command COPY-USERS-FILES(Documentation on all of the possible function names will be found in the SINTRAN III Utilities Manual, ND-60.151):

@BACKUP-SYSTEM

BA-SY:COPY-USERS-FILES < destination >,< source >

For example, to copy all of your files of type :TEXT onto a magnetic tape:

@BACKUP-SYSTEM

BA-SY:COPY-US-FI

DESTINATION TYPE: VOL

DEST. VOLUME-NAME: BACKUP

DEST. DEVICE-NAME: MAG-TAPE-1

DEST. UNIT-NUMBER: 0

DEST. FILE-GENERATION: 1

SOURCE TYPE: DIR

SOURCE DIRECTORY-NAME: PACK-ONE

SOURCE USER-NAME: JOHN-SMITH

SOURCE FILE-NAME: :TEXT

MANUAL CHECK: Y

BA-SY:EX

@

The first parameter (VOL) indicates the type of the destination, here it is a volume.

The second parameter (BACKUP) is the name of the volume. (The system will make sure that the magnetic tape on the designated unit has the name you specify).

The third parameter (MAG-TAPE-1) is the device on which the magnetic tape is mounted.

The fourth parameter (0) is the unit number on that device.

The fifth parameter (1) is the generation number, which allows several different copies of the same files to be kept on one volume.

The sixth parameter (DIR) starts the description of the source. Here it is a directory.

The seventh parameter (PACK-ONE) is the name of the directory.

The eighth parameter (JOHN-SMITH) is the name of the user whose files are to be copied.

The ninth parameter (:TEXT) names all the files you wish to be copied, in this case all files of the type TEXT.

The tenth parameter (Y(es)) means that for each file to be copied, the file name will be written on the terminal and the program will wait until you indicate whether the file really should be copied (press Y and CR) or whether it should be skipped (press N and CR). If you have finished copying and you still have many files to list, you can escape from this mode by pressing escape.

To recover these files one may write:

```
BA-SY:C-U-F DIR,PACK-ONE,JOHN-SMITH,  
VOL,BACKUP,MAG-TAPE-1,0,1,:TEXT,N
```

where the first three parameters describe the destination, and the remainder describe the source.

Note: Volumes are created on new tapes by the command:

```
BA-SY:CREATE-VOLUME < volume name >,< device name >,< unit >
```

Floppies are best used for maintaining copies of a few crucial files.

To copy a single file, you can use the COPY-FILE command in SINTRAN:

```
@COPY-FILE < destination file >,< source file >
```

For example:

```
@COP-FIL (FLOPPY:USER)FILEA:DATA,FILEA:DATA
```

which will copy your file called FILEA:DATA onto the floppy disk. Note that the floppy disk contains a normal directory and that you are conforming to the normal file-naming conventions.

Note: Before you use a floppy you must use the command ENTER-DIRECTORY. Similarly, when you have finished using the floppy you must use the command RELEASE-DIRECTORY.

4. Sintran Information

4.1. General

This chapter describes commands and monitor calls for measuring the performance and obtaining information about programs.

Commands for listing file system information is described in section 3.10. Commands to give performance information are found in the SINTRAN III Real Time Guide (ND 60.133).

4.2. Miscellaneous

4.2.1. @LIST-REENTRANT

List the reentrant subsystems established by the @DUMP-REENTRANT command (user SYSTEM). For example:

```
@LIST-REENTRANT
START RESTART SEGMENT
      0      1      112    QED
177777 177775      114    MAC
      ...
```

4.2.2. @LIST-TITLE

List the system identification normally displayed when logging on, e.g.:

```
@LIST-TITLE
SINTRAN III ~ VS VERSION F
@
```

4.2.3. @HELP

@HELP <command>,<output file>

List command names matching <command>, e.g.:

```
@HELP EX,,
RT: EXECUTE-IOX
FILSYS: EXPAND-FILE
@
```

Two commands start with "EX", one RT and one file-system command.

4.2.4. @TERMINAL-STATUS

See section 2.6.

4.2.5. @WHO-IS-ON

See section 2.6.

4.3. The SINTRAN III Calendar

ND computers are equipped with a clock. The date and time of day are stored in SINTRAN III in basic time units. One basic time unit is normally 20 milliseconds. The calendar can be read either in basic time units or to an array in decoded form.

4.3.1. @DATCL

See section 2.6.

4.3.2. @TIME-USED

See section 2.6.

4.3.3. CLOCK (MON 113)

Read the current setting of date and time of day, e.g.:

```
INTEGER CLDAT(7)
CALL CLOCK(CLDAT)
```

CLDAT will receive the following data:

CLDAT(1) : basic time units

CLDAT(2) : seconds

CLDAT(3) : minutes

CLDAT(4) : hours

CLDAT(5) : day

CLDAT(6) : month

CLDAT(7) : year

4.3.4. TIME (MON 11)

Read current internal time, e.g.:

```
DOUBLE INTEGER TIME,TI
```

TI=TIME(0)

TI will receive the internal time in basic time units.

4.3.5. TUSED (MON 114)

Read the CPU time used since the user logged on or the batch job was started. For example:

DOUBLE INTEGER TD,TUSED
TD=TUSED(0)

TD will receive the CPU time used in basic time units.

5. Executing User Programs and Subsystems

5.1. General

When creating and executing programs under SINTRAN III, the symbolic code is created as a SYMB-file. By compiling it, a BRF-file is created, containing relocatable code. The ND Relocating Loader (NRL) is used to load and execute the program.

Once the program is loaded, SINTRAN III has available commands for

- creating a PROG-file. This is the program in reloadable format.
- loading, starting, and restarting a program.
- examining the user's registers and memory.
- loading binary programs. A binary program is loaded from a BPUN-file. It is similar to a PROG-file except that it can also be loaded as a stand-alone program (i.e., independently of SINTRAN III).

Also, there are monitor calls available for extending the address space from 64K to 128K words.

Time-sharing users have available a command for listing programs defined as reentrant subsystems, @LIST-REENTRANT. It is described in chapter 4.

5.2. Creating a PROG-File

When a program has been loaded by the ND Relocating Loader the program area in the user's memory may be dumped to a PROG-file. The file will contain the program in reloadable format.

5.2.1. @MEMORY

@MEMORY <low address>,<high address>

Define the area of the user's memory to be dumped (by @DUMP, see section 5.2.2.). An example:

@MEMORY 10,1777

Set dump limits to 10 to 1777 (octal), limits inclusive.

5.2.2. @DUMP

@DUMP <file name>,<start address>,<restart address>

Dump the user memory area (as defined by @MEMORY) to a file. When the program is started by @RECOVER, the execution will begin at <start address>. When restarted by @CONTINUE, the execution will begin at <restart address>. For example:

@MEMORY 10,1777
@DUMP MYPROG,10,11

The file MYPROG:PROG in the user's default directory will contain a dump of the user area 10-1777. Program start address is 10 and restart address is 11. (All values in octal.)

5.3. Controlling Execution

This section describes commands for starting user programs and subsystems. A general description of how to execute programs is found in section 2.3. The commands described here are used for the transfer between Command Mode and User Mode as shown in figure 4. The command type determines the start address to be used.

5.3.1. @RECOVER

@RECOVER <program name>

Load a PROG-file to the user's memory and start it at the <start address> as specified by @DUMP or @DUMP-REENTRANT. (@DUMP-REENTRANT is explained in the SINTRAN III System Supervisor Manual.) @RECOVER is a default command, i.e., if only the program name is specified, the command is assumed to be @RECOVER. For example:

@RECOVER MAC

where MAC is a SINTRAN III subsystem, is equivalent to

@MAC

@RECOVER MYPROG

where MYPROG:PROG is a user defined program, is equivalent to

@MYPROG

When SINTRAN III searches for a file name or program name, it first searches the users files.

USER is searched:

IF <file name> is:	THEN:
- found and not ambiguous -----	File is fetched.
- ambiguous -----	User SYSTEM is searched.
- not found -----	User SYSTEM is searched.

User SYSTEM is searched:

IF <file name> is:	THEN:
- found and not ambiguous -----	File is fetched.
- ambiguous -----	Error message 'AMBIGUOUS FILE NAME'.
- not found -----	Error message 'NO SUCH FILE NAME' (if not found when user was searched) or 'AMBIGUOUS FILE NAME'(if ambiguous when user was searched).

E.g.:

The user types:

@FORT

If the user has no such file name, SINTRAN III will search user SYSTEM's files. User SYSTEM has one file FORTRAN-OLD:PROG and one file FORTRAN-82060B:PROG. The error message will then be:
AMBIGUOUS FILE NAME
since FORT can be any of the two.

5.3.2. @CONTINUE

Restart a program at the <restart address> as specified by @DUMP. (The program is not loaded, only restarted.)

5.3.3. @GOTO-USER

@GOTO-USER <address>

Note: Address indicating the memory location number (=P-register address, see example below).

Start a program at a specific address. The default address is the point at which the program was interrupted, e.g.:

(The user presses ESCAPE during execution.)

@STATUS

P= 1723

X= 5

...

@GOTO

The program is restarted in address 1723 octal. (@STATUS is explained in the next section).

5.4. Examining User's Registers and Memory

User's registers and memory may be both examined and modified by the following commands (all figures are octal):

5.4.1. @STATUS

Display the contents of the user's registers, e.g.:

```
@STATUS
P= 3157          %Program counter
X= 13302         %X register
T= 105           %T register
A= 12663         %A register
D= 47            %D register
L= 605           %L register
S= 40            %Status register
B= 101           %B register
@
```

5.4.2. @LOOK-AT@LOOK-AT <space reference>

This is a general command for modifying user's registers and memory. The <space reference> permitted for time-sharing users are

MEMORY - the time-sharing user's virtual memory (0-64K)

ALT-MEMORY - the time-sharing user's alternative virtual memory. The addresses are specified relative to the 64K boundary.

The references SEGMENT, RTCOMMON and REGISTER are restricted to users RT and SYSTEM, the references IMAGE and RESIDENT only to user SYSTEM while ALT-MEMORY is available to all users. Addresses and registers are specified on the line following the command. Typing . (dot) causes return to command mode, e.g.:

```
@LOOK-AT REG
X/ 13302 13000
.
- -END
@
```

Change the contents of the X register from 13302 to 13000.

```
@LOOK-AT MEM
13000/ 10 100
11 .
@
```

Change the contents of location 13000 from 10 to 100. (11 is the contents of location 13001.)

5.4.3. @SET-MEMORY-CONTENTS

@SET-MEMORY-CONTENTS <contents>,<low address>,<high address>

Set all locations of an area of the user's memory to the same value. This is more convenient than using @LOOK-AT MEMORY, e.g.:

@SET-MEMORY-CONTENTS 124000,0,177777

Set all locations of the user's memory to 124000 (i.e. the instruction JMP *).

5.5. Loading Binary Programs

The following two commands are used for loading programs created by the)BPUN command in MAC or by the BPUN command in NRL :

5.5.1. @LOAD-BINARY

@LOAD-BINARY <file name>

Load a program in BPUN format into the user's memory and start executing it. The command is similar to @RECOVER except that the program file is in BPUN format. For example:

@LOAD-BINARY PROG-1

The program in the file PROG-1:BPUN is loaded and executed.

5.5.2. @PLACE-BINARY

@PLACE-BINARY <file name>

Load a program in BPUN format into the user's memory. The program is not started, e.g.:

@PLACE-BINARY PROG-1

The program in the file PROG-1:BPUN is loaded.

5.6. Error Messages from User Programs

Section 2.5 gives an overview of error handling in SINTRAN III. By selecting one of two monitor calls the user can decide whether an error should just be displayed or whether it should also cause program termination.

5.6.1. ERMSG (MON 64)

Display the error message for a given error number. This monitor call is mostly used in connection with the error return of some other call, e.g.:

```
MON 50      %OPEN
MON 64      %ERMSG, error return
...        %normal return
```

After displaying the error message the execution resumes at the next location.

5.6.2. QERMS (MON 65)

Display the error message for a given error number and exit from the program. The monitor call is used similarly to ERMSG:

```
MON 50      %OPEN
MON 65      %QERMS, error return
...        %normal return
```

After printing the error message the program exits.

5.7. Communicating with One's Own Terminal

LDN = 0 specifies your terminal in input/output monitor calls. On input the text line is transferred to the SINTRAN III input buffer on receiving a carriage return. While the line is being entered, the QED control characters are available. In order to use the 'old line' for editing, the user program must first read the line and then use SETCM to transfer the line back to the input buffer.

By using LDN=0 the program may start with reading the rest of the @RECOVER command. This is carried out with a REWIND statement. Assume the @RECOVER command is:

@USPROG TEXT

where USPROG is the name of the user program and "TEXT" is text to be read by the program. USPROG must then look as follows:

```
REWIND 0
READ(0,1)AA
10     FORMAT(A16)
```

Finally, a SINTRAN III command may be executed by specifying it as a text string (COMND).

5.7.1. SETCM (MON 12)

Transfer a string to the command input buffer. It is used for 'old line' editing.

5.7.2. COMND (MON 70)

Execute a character string as a SINTRAN III command, e.g.:

```
CHARACTER DELFI*50  
DATA DELFI/'DELETE-FILE XXX:SYMB'''/  
...  
CALL COMND(DELFI)
```

The command:

@DELETE-FILE XXX:SYMB

is executed. If an error occurs, the program is aborted.

5.7.3. Terminating Programs

Normally only the monitor call LEAVE is used for program termination, but batch jobs need two methods of termination. Either the user program is to be terminated or, if there is a serious error, the whole job should be terminated. RTEXT is used in the latter case.

5.7.4. LEAVE (MON 0)

Terminate the executing program and return control to the operating system. In batch jobs, the next command of the job is executed.

5.7.5. RTEXT (MON 134)

Terminate the executing program and return control to the operating system. In batch jobs the job is terminated.

6. Terminal Characteristics

6.1. General

With the wide variety of computer terminals on the market, it is necessary for SINTRAN III to treat the different models individually.

This treatment includes adapting to the communication characteristics (@TERMINAL-MODE, @DISABLE-ESCAPE-FUNCTION, @ENABLE-ESCAPE-FUNCTION) and identifying the model (@SET-TERMINAL-TYPE, @GET-TERMINAL-TYPE).

In addition, the system supervisor has available commands for selecting which terminal is to receive the system error messages (@SET-ERROR-DEVICE, @GET-ERROR-DEVICE) and commands for selecting the value of the ESCAPE (ESC) character (@DEFINE-ESCAPE-CHARACTER).

These commands are explained in the SINTRAN III System Supervisor manual, ND.60.103.

The terminal type is identified by a 16 bit terminal number associated with the LDN. The current numbers defined by Norsk Data A.S are shown in appendix B of the SINTRAN III Reference Manual. If a terminal is sold as part of the system it is also listed with its ND-number.

If a user wishes to standardize a particular terminal, he should contact ND to ask it to be entered into the list.

6.2. Terminal Communication

6.2.1. @TERMINAL-MODE

This command sets the communication mode of the terminal. The mode determines how SINTRAN III should handle the terminal. The functions are:

- converting lower case alphabetic characters to upper case on input.
- causing delay after CARRIAGE RETURN (CR) on output by printing filler (dummy) characters.
- stopping after 20 lines of output in order to make it possible to read fast displays.
- logging out the terminal on detecting that the carrier is missing. (RS 232/V24 lines only)

The command parameters are yes or no in answer to questions. Default values cause no change in the function, e.g.:

```
@TERMINAL-MODE
CAPITAL LETTERS?N
DELAY AFTER CR?
STOP ON FULL PAGE?Y
LOGOUT ON MISSING CARRIER?
@
```

Capital letters should not be converted. Output should stop after 20 lines of output. The other functions are not changed.

6.2.2. @DISABLE-ESCAPE-FUNCTION

@DISABLE-ESCAPE-FUNCTION <LDN>

Disable the function of the ESCAPE character. It is disabled until the command @ENABLE-ESCAPE-FUNCTION is given or the user logs out. Time-sharing users are only permitted to use the default value of <LDN> which is their own terminal. (Only user SYSTEM can specify a terminal other than his own.)

6.2.3. @ENABLE-ESCAPE-FUNCTION

@ENABLE-ESCAPE-FUNCTION <LDN>

Enable the function of the ESCAPE character. Time-sharing users are only permitted to use the default value of the parameter which is their own terminal. (Only user SYSTEM can specify a terminal other than his own.)

6.2.4. TERMO (MON 52)

Set terminal mode for any terminal. The call corresponds to @TERMINAL-MODE, e.g., in a background program:

```
INTEGER TERMO
ISTAT=TERMO(0,4)
```

Set "stop on full page" for user's terminal. All other modes are set to NO.

6.2.5. DESCF (MON 71)

Disable escape function. The call corresponds to @DISABLE-ESCAPE-FUNCTION above. For example:

```
CALL DESCF(ITERM)
```

Disable the escape function for one's own terminal. The LDN in ITERM is ignored for background programs.

6.2.6. EESCF (MON 72)

Enable escape function. It corresponds to @ENABLE-ESCAPE-FUNCTION above. For example:

```
CALL EESCF(ITERM)
```

ITERM is ignored for background programs.

6.2.7. ECHOM (MON 3)

Set the echo strategy of a terminal. A description of this strategy is found in SINTRAN III SYSTEM DOCUMENTATION, section 3.4.6.4 (ND-60.062), e.g.:

```
LDN=9  
CALL ECHOM(LDN,1)
```

On terminal 2 set echo on all characters except control characters.

6.2.8. BRKM (MON 4)

Set the break strategy of a terminal. A description of this strategy is found in SINTRAN III SYSTEM DOCUMENTATION, section 3.5.6.4 (ND-60.062), e.g.:

```
LDN=9  
CALL BRKM(LDN,0)
```

On terminal 2 set break on all characters.

6.3. Terminal Identification

6.3.1. @SET-TERMINAL-TYPE

@SET-TERMINAL-TYPE <LDN>,<terminal number>

Associate a terminal-type number with a terminal. Time-sharing users are only permitted to use the default value of the parameter which is their own terminal, e.g.:

@SET-TERMINAL-TYPE ,160007B

The terminal is identified as TANDBERG-TDV2000.

6.3.2. @GET-TERMINAL-TYPE

@GET-TERMINAL-TYPE <LDN>

Display the terminal type or type number. (Time-sharing users may use any LDN.) For example:

@GET-TERMINAL-TYPE 52
TERMINAL TYPE : -8189

@

The terminal number of LDN = 52 (decimal) is displayed as a decimal signed integer.

6.3.3. MSTTY (MON 17)

Set the terminal type, e.g.:

CALL MSTTY(0,4)

Set the terminal number of the terminal to 4.

6.3.4. MGTTY (MON 16)

Get the terminal number, e.g.:

CALL MGTTY(0,ITY)

The terminal type of this terminal is stored in ITY.

6.4. Suspending the Terminal

6.4.1. @HOLD

@HOLD <no. of time units>,<time unit>

Keep the terminal waiting for a specified amount of time, e.g.:

@HOLD 5,3

@

The last @ occurs after 5 minutes.

6.4.2. HOLD (MON 104)

Set the calling program in a wait state for a specified amount of time. For example:

CALL HOLD(10,2)

The calling program will wait for 10 seconds before continuing with the next statement.

7. Batch- and Mode-Jobs

7.1. General

SINTRAN III commands can be processed interactively on the terminal by the user, as a mode-job or as a batch-job independent of any terminal.

A mode-job allows a user to submit a set of commands and other input responses previously stored (normally on a mass storage file) to be executed automatically. Running a mode file occupies the terminal to which command output can be routed. A batch-job (also on a mass storage file, or from a card-reader) runs independently of the terminal with file and command output routed to some other file or line-printer.

The SINTRAN III batch processing system comprises one or more batch processors. The number is determined at system generation time. Internally, each processor is implemented as a background program similar to the ones that run the terminals.

User SYSTEM has available commands for starting and stopping each batch processor (@BATCH and @ABORT-BATCH).

A batch processor may be in one of three states:

PASSIVE: the batch processor has not been started.

IDLE: the batch processor has been started but its queue is empty.

ACTIVE: the batch processor is working on a batch job.

In the IDLE state, when any user appends a batch file to a queue, the processor will be activated. Batch files will be taken from the queue until it becomes empty. The processor then goes back to the idle state.

7.2. Definitions

A batch job is a collection of SINTRAN III and subsystem commands, arranged in the sequence they would be entered from a terminal keyboard. The first command of the job must be @ENTER. It has the function of "logging on" a batch user. The batch job will then "belong to" this user. The commands in the job will be regarded as coming from this user for file access and accounting. Double escape is logging the batch job off (Entered by control 0 and escape (<0>esc)).

The following are not permitted in batch jobs:

- running a mode-job(@MODE).
- use TERMINAL as output file.
- the command @LOGOUT.
- append a batch job(@APPEND-BATCH)

A mode-job is equivalent to a batch-job, only it is used in a different way. The mode-job is started by the @MODE command and the terminal is dedicated to the mode processing until all commands are executed. Any @ENTER-command is ignored. The mode-job belongs to the user giving the @MODE command.

A job input file is either a batch input file or a mode input file. It contains one or more jobs to be processed. The boundaries between the jobs are determined by @ENTER or the end of the job input file. @ENTER are used for batch only.

A job output file is either a batch output file or mode output file. It is the destination of the command output normally displayed on the terminal. A batch job cannot have TERMINAL as output file.

A batch queue is a list of batch files to be input to the batch processor. Each entry in the queue consists of a batch input file and a corresponding batch output file.

7.3. Sample Batch File

The example below will do the following:

- Compile a FORTRAN program included in the job.
- Load the program together with subprograms and library routines.
- Execute the program with data from the batch file.

```
@ENTER USER-ONE,1814,100,2
@CC      *****
@CC      ***  SAMPLE BATCH FILE  ***
@CC      *****
@FTN
COM 1,1,PROGA
      PROGRAM A
      I1=0
      I2=0
      LL=10
      ...
      END
      EOF
EX
@NRL
LOAD TEST,SUBLIB
LOAD FTNLIBR
ENTRIES-DEFINED
```

```
ENTRIES-UNDEFINED
DUMP PROGA
EXIT
@PROGA
10 20 30
11 15 -1
```

<O>esc<O>esc

In the example above, 1814 is the password of USER-ONE, 100 is the project password and 2 is the maximum no. of minutes for the job. Note that SINTRAN III commands must include the attention character (@). The subcommands must not have the attention character prefixed unless when SINTRAN-SERVICE-PROGRAM or MAIL commands.

The job can also be run as a mode-job. The @ENTER-command will then be ignored. It must be run by USER-ONE only if files accessed are peculiar to USER-ONE.

7.4. Commands for Running Batch Jobs

These are commands for appending a batch file, aborting the currently running job and deleting an entry from the batch queue. It is also possible to list the contents of a queue and the status of the batch processor.

7.4.1. @APPEND-BATCH

@APPEND-BATCH <batch no.>,<input file>,<output file>

Append a batch file to the queue of a batch processor, e.g.:

@APPEND-BATCH 1,JOB-1,LINE-PRINTER

The batch-file JOB-1:SYMB is appended to the batch processor. Output will be appended to the LINE-PRINTER.

7.4.2. @ABORT-JOB

@ABORT-JOB <batch no.>,<user name>

Abort the current batch job being processed. The next job will be initiated. A time-sharing user can only abort a job if it belongs to his own user name. For example:

(user giving the command is GUEST)

@ABORT-JOB 1,GUEST

The current batch job for batch processor 1 is aborted provided its owner is GUEST.

7.4.3. @DELETE-BATCH-QUEUE-ENTRY

@DELETE-BATCH-QUEUE-ENTRY <batch no.>,<input file>,<output file>

Delete a batch file in the batch queue. A time-sharing user can only delete a file belonging to himself. <input file> and <output file> must be spelled the same way as in the corresponding @APPEND-BATCH command. (Use @LIST-BATCH-QUEUE to find the correct spelling.) For example:

@DELETE-BATCH-QUEUE-ENTRY 1,JOB-1,LINE-PRINTER

The batch file JOB-1 is deleted from the queue of batch processor 1, provided the second and third parameters match exactly an entry in the queue.

7.4.4. @LIST-BATCH-QUEUE

@LIST-BATCH-QUEUE <batch no.>

List the contents of the batch queue, e.g.:

@LIST-BATCH-QUEUE 1

1 CARD-READER LINE-PRINTER
2 (USER-NAME)JOB-1 LINE-PRINTER
@

There are two entries in the queue. The file "CARD-READER" is the next one to be processed.

7.4.5. @LIST-BATCH-PROCESS

List the state of each batch process defined in the system, e.g.:

@LIST-BATCH-PROCESS

1 IDLE, NO USER LOGGED ON
2 ACTIVE USER GUEST LOGGED ON
3 PASSIVE

@

Three processors are defined. The second one runs a job belonging to user GUEST.

7.5. Commands for Running Mode-Jobs

7.5.1. @MODE

@MODE <input file>,<output file>

Substitute the current command input (normally TERMINAL) with commands from the <input file>. Command output is routed to the <output file> (normally TERMINAL). When receiving end-of-file on the <input file> the command input and output are routed to the previous destinations, e.g.:

(user is logged on as USER-ONE)

@MODE JOB-1,TERM

... (command output)
@

If the command is issued interactively at the terminal, the next commands are taken from the mass storage file "JOB-1:SYMB". The command output is routed to the terminal.

(the following command resides in the file JOB-1:SYMB)

@MODE JOB-2,TERM

The next commands will be taken from JOB-2:SYMB and the output is routed to the <output file> specified by the @MODE-command previously given at the terminal. This is the <output file> at the "top" of the @MODE nesting and need not necessarily be TERMINAL.

7.6. Commands within Mode- or Batch-Jobs

The command @ENTER has already been mentioned as the first command in a batch file. @SCHEDULE is used to reserve devices.

Each SINTRAN III command must start with the attention character @. Subcommands, except for SINTRAN -SERVICE-PROGRAM and MAIL, do not use any attention character. All command parameters must be specified on the same line. @LOGOUT is not permitted within batch jobs. Also, some commands are only relevant for interactive use. Mode-jobs may contain @MODE commands for nesting jobs.

If an error occurs in a job, the error message is written to the job output file together with the message

*** BATCH JOB ABORTED ***

The job is then aborted. If an error occurs in accessing the job input or output file, an error message is routed to the error device. This applies to batch only.

7.6.1. @ENTER

@ENTER <user name>,<password>,<project password>,<max. time>

Start a new batch job. The parameters are similar to the ones used when logging on. The job is terminated after the last command or by being aborted when <max. time> has expired. An example of @ENTER was shown in the sample program of section 7.3.

7.6.2. @SCHEDULE

@SCHEDULE <logical dev. no.>,<logical dev. no.>, ...,
<logical dev. no.>

Reserve a set of devices for the batch job. If any device is already reserved, the batch processor will enter a waiting state until all devices are released, e.g.:

@SCHEDULE 2,5

Reserve the tape-reader and line-printer.

7.6.3. @CC

@CC <text>

This command functions as a comment in batch or mode jobs. Section 7.3 shows an example of three @CC commands.

7.7. Monitor Calls for Mode- and Batch-Jobs

7.7.1. RSIO (MON 143)

This is a call for determining the execution mode of the calling program. The modes are: Interactive, batch-job or mode-job, e.g.:

CALL RSIO(IEXEC,IINP,IOUTP,IUSER)

IEXEC will on return contain the execution mode which is 0 for interactive, 1 for batch and 2 for mode. IINP, IOUTP and IUSER will contain the file number of the input job file, the output job file and the user number, respectively. (The user number is the value listed by @LIST-USERS.)

8. Spooling

8.1. General

The term spooling is derived from the acronym SPOOL which stands for Simultaneous Peripheral Output On Line. In SINTRAN III it means that files to be printed can be put in a queue waiting to be output. This is obviously more convenient for the user than having to wait for the printer to be available, then reserving it, copy the file (@COPY-FILE, etc.), and finally releasing the printer. Files are being output from the spooling queue as an independent process in the computer.

A peripheral with the spooling feature will have a peripheral file with more than one version. Version 1 is the normal peripheral file. The other files are mass storage files, called peripheral spooling files (PSF), or usually just spooling files. They will receive the data when the user writes to the peripheral (@COPY, @COPY-FILE, etc.). After data is written the PSF it is appended to the spooling queue. The user can also append a user file directly to the spooling queue by means of the command @APPEND-SPOOLING-FILE. This is faster than a copy command since it bypasses the PSF, but the user cannot access the file while it is being printed. Thus, the spooling queue may contain two types of files, PSFs and users files.

Normally the user wants the output to be started as soon as his file becomes the first one in the queue and the preceeding file is finished. However, the user may also specify a stop condition, i.e., the output will stop when the file is ready to be printed. A user message will appear on the error device and the output must be started by a @START-PRINT command. Only user SYSTEM and the user who appended the file to the queue are permitted to issue this command. The stop condition is useful for printing on special forms.

In general there are three ways of specifying a stop condition.

1. User SYSTEM may specify stop condition for all files in the queue. (@DEFINE-SPOOLING-CONDITIONS)
2. A time-sharing user may specify a stop condition for all PSFs he will append. (@SPOOLING-FILE-MESSAGE)
3. A time-sharing user may specify a stop condition for the particular file he appends. (@APPEND-SPOOLING-FILE)

The output may also be stopped while a file is being printed (@STOP-PRINT).

Files are normally printed in the order in which they occur in the queue. User SYSTEM may specify that only files with a specific user message should be printed (@SET-SPOOLING-FORM). The other files will not be printed until the SPOOLING-FORM identification is changed to match the actual user text. Time-sharing users may use @LIST-SPOOLING-FORM to check the message.

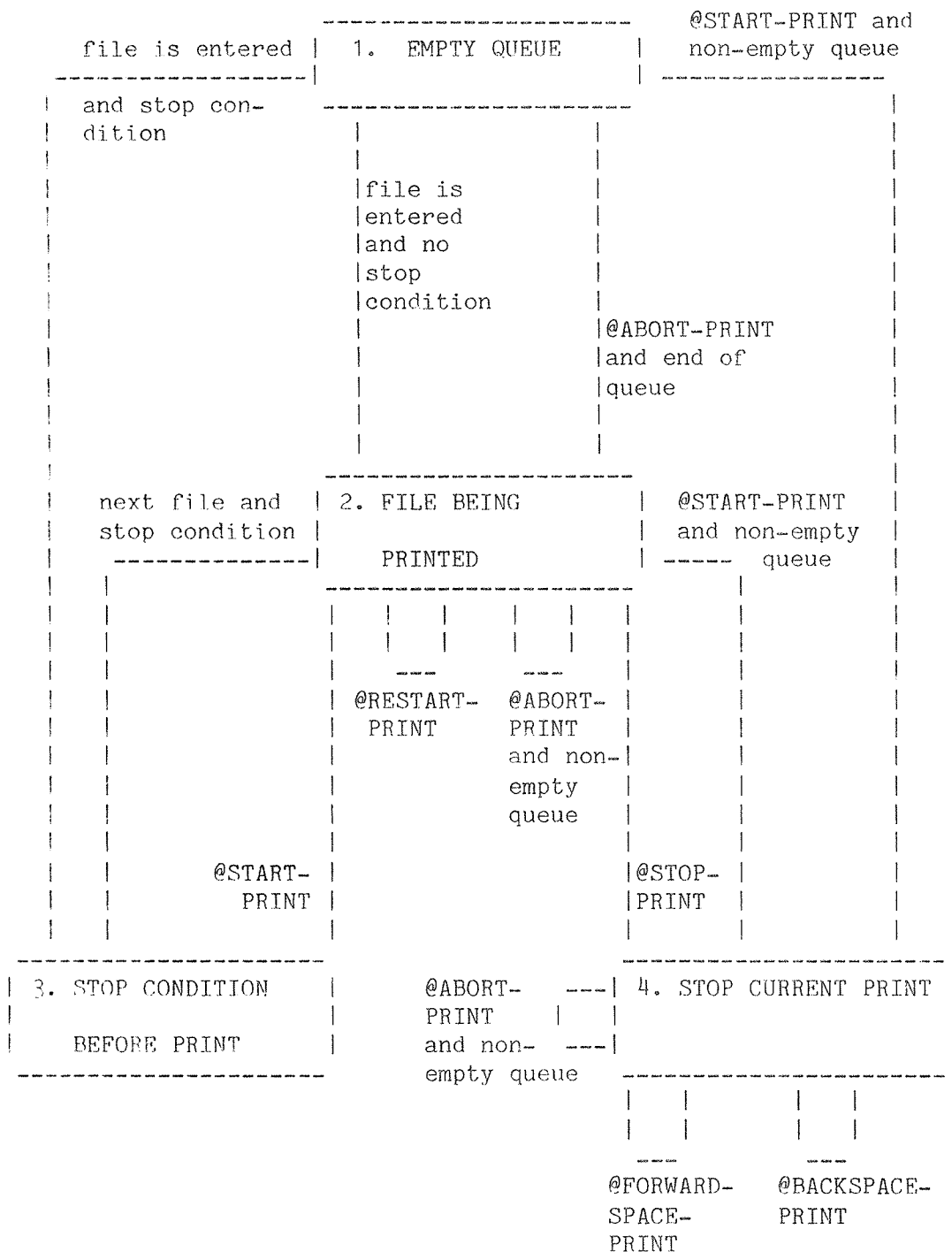


Fig. 14. Spooling System for Time-sharing Users

Figure 14 shows the different states of the spooling system while it is activated by user SYSTEM (@START-SPOOLING). In state 1 there are no files in the queue. (The message from @LIST-SPOOLING-QUEUE is 'QUEUE IS EMPTY'.) The system will leave this state as soon as a file is put in the queue. Normally no stop condition is specified and the system is transferred to state 2 where the output starts immediately. (@LIST-

SPOOLING-QUEUE now returns the message 'FILE CURRENTLY PRINTED IS ...'. The system will be in this state as long as the queue contains files to be printed without stop condition. When the queue becomes empty the system returns to state 1.

In state 2 the file may be aborted (@ABORT-PRINT) causing the print to be terminated and the next file to be started, if any. The file may also be restarted (@RESTART-PRINT) causing the print to start over again from the first page.

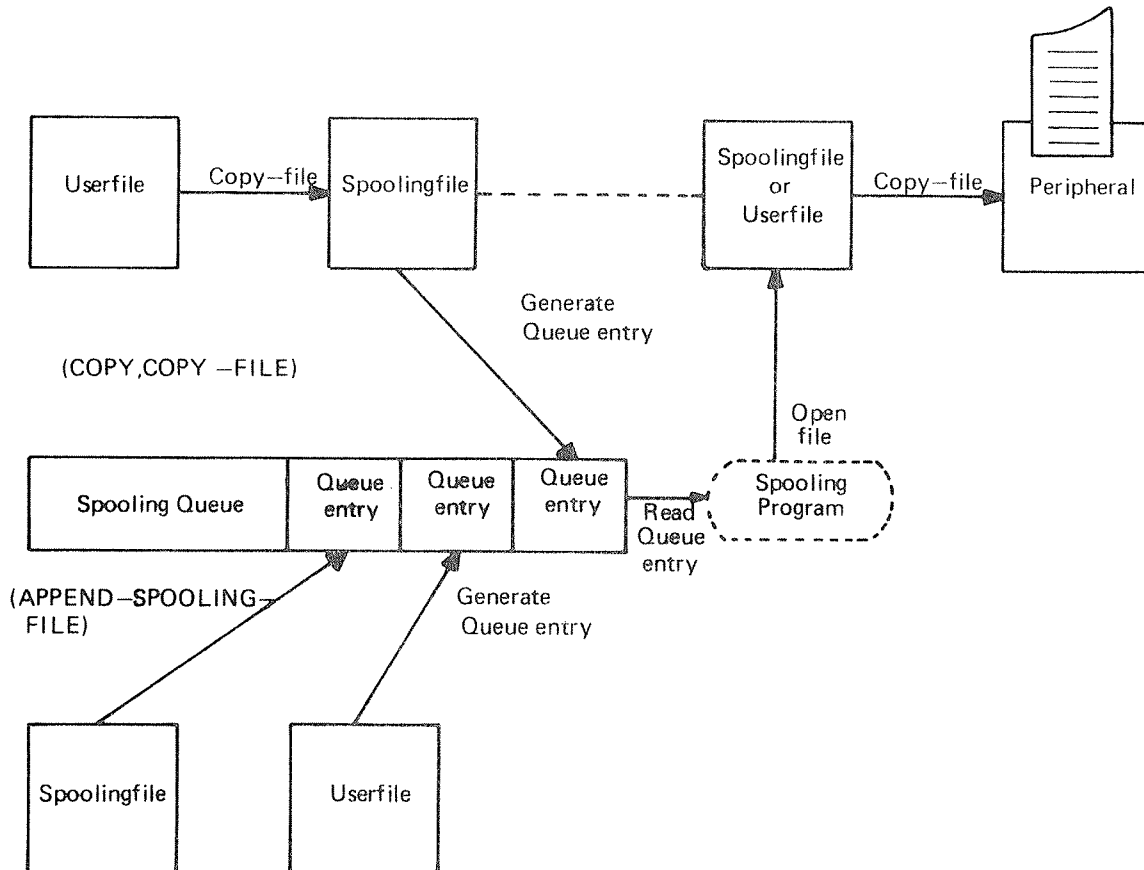


Fig. 15. Spooling System, Queue Entries

A file with a stop condition will cause the system to enter state 3 when it is ready to be printed. (This state is indicated by a user message on the error device.) Note that in this state the file is still part of the queue. On the @START-PRINT command the system is transferred to state 2 where the file is taken from the queue and printed. Special care must be taken when deleting the file in state 3. Since it is still part of the queue, @ABORT-PRINT and @FORWARD-SPACE-PRINT do not work. The file must be removed by @DELETE-SPOOLING-FILE. The user should note that the spooling must then be started either by user SYSTEM or the user who appended the deleted file and not the one who appended the next file in the queue. In state 2, @STOP-PRINT transfers the system to state 4, i.e., stop current print. (It can be verified by @LIST-SPOOLING-QUEUE. The message 'FILE CURRENTLY PRINTED

...' is deleted.) Stopping the print is useful should something unexpected occur during the output (paper crash etc.). @RESTART-PRINT, @ABORT-PRINT, @FORWARD-SPACE-PRINT, and @BACKSPACE-PRINT is accepted, but no output occurs until the next @START-PRINT. If the file is aborted the print must be started by user SYSTEM or the user who appended the aborted file, similarly to the rule for state 3.

Time-sharing users will normally only need to copy or append a file to the queue and maybe check the proceedings of the output. The basic commands used for this purpose is @COPY-FILE, @APPEND-SPOOLING and @LIST-SPOOLING-FILE.

8.2. Appending and Changing the Queue Entry

As mentioned previously, a queue entry may be a PSF or a user file. The queue entry contains information about the file name, the number of copies to be printed, and, if used, a text to be printed out on the error device before the user starts the printout.

8.2.1. @APPEND-SPOOLING-FILE

@APPEND-SPOOLING-FILE <peripheral file name>,<file name>,
<no. of copies>,<text>,[<PRINTING MESSAGE
INDEPENDENTLY OF SPOOLING CONDITIONS?>]

Append one or more copies of a file to a spooling queue, e.g.:

@APPEND-SPOOLING-FILE LINE-PRINTER,OLE,,,'

The user file OLE is put in the spooling queue to line-printer. It is started directly (state 2) unless the user has specified a stop condition.

@APPEND-SPOOLING-FILE LINE-PRINTER,OLE,,OLE IS READY',Y

The command is similar to the one above except that when the file is ready to be output, the message 'OLE IS READY' is output to the error device (state 3). The output is independent of any stop condition.

8.2.2. @SET-NUMBER-OF-PRINT-COPIES

@SET-NUMBER-OF-PRINT-COPIES <peripheral file name>,<file name>,
<no. of copies>

Change the number of copies to be printed for a specific file in the queue. For example:

@SET-NUMBER-OF-PRINT-COPIES LINE-PRINTER,OLE,2

Two copies of the file OLE will be printed.

8.2.3. @DEFINE-SPOOLING-FILE-MESSAGE

@DEFINE-SPOOLING-FILE-MESSAGE <text>',

<PRINTING MESSAGE INDEPENDENTLY OF SPOOLING CONDITIONS?>

Associate a text with a spooling-close. This means that if a user file is copied to a PSF the text will be attached to the peripheral spooling file. When the file is ready to be printed, the text is output to the error device and the spooling system will wait for a @START-PRINT, e.g.:

@DEFINE-SPOOLING-FILE-MESSAGE READY FOR USER GUEST',Y

Print the text 'READY FOR USER GUEST' and wait for @START-PRINT before printing any PSF containing text from the users own files.

@DEFINE-SPOOLING-FILE-MESSAGE ',Y

Reset to normal operation.

8.3. Changing the Order in the Queue

The time-sharing user is permitted to remove one of his files from the queue or to move it backwards. User SYSTEM may also select a subset of the files in the queue to be printed (@SET-SPOOLING-FORM).

8.3.1. @MOVE-SPOOLING-QUEUE-ENTRY

@MOVE-SPOOLING-QUEUE-ENTRY <peripheral file name>,<file name>,
<INSERT OR APPEND?>,<before/after file name>

Change the order of a file in the spooling queue. Only user SYSTEM may move entries forward in the queue, e.g.:

@MOVE-SPOOLING-QUEUE-ENTRY LINE-PRINTER,F-1,I,F-2

The file F-1 is moved in front of F-2 in the spooling queue.

8.3.2. @REMOVE-FROM-SPOOLING-QUEUE

@REMOVE-FROM-SPOOLING-QUEUE <peripheral file name>,<file name>

Remove a file from the spooling queue. The contents of the file are retained, e.g.:

@REMOVE-FROM-SPOOLING-QUEUE LINE-PRINTER,F-1

If found in the spooling queue to LINE-PRINTER, the file F-1 is removed.

8.3.3. @DELETE-SPOOLING-FILE

@DELETE-SPOOLING-FILE <peripheral file name>,<file name>

Remove a file from the spooling queue. If it is a PSF its pages are released and returned to the pool of free spooling pages, e.g.:

@DELETE-SPOOLING-FILE LINE-PRINTER,LINE-PRINTER::10

If found in the spooling queue to LINE-PRINTER, the PSF LINE-PRINTER::10 is removed and its pages deleted. It will then exist as a mass storage file with zero pages.

8.4. Starting and Stopping the Print

The time-sharing user may start and stop the printing of his own files or the PSFs that he has used.

8.4.1. @START-PRINT

@START-PRINT <peripheral file name>

Continue the printing of the current file, or, if there is no current file, start the printing of the next file.

8.4.2. @STOP-PRINT

@STOP-PRINT <peripheral file name>

Stop the current printout. The command may take some time to take effect because of data buffering

8.4.3. @RESTART-PRINT

@RESTART-PRINT <peripheral file name>

Restart the file currently being printed on the spooling device. If the printing is stopped, the command will only reset the file. (It must be started by @START-PRINT).

8.4.4. @ABORT-PRINT

@ABORT-PRINT <peripheral file name>

Abort the current printout. A time-sharing user may only abort the printing of a file appended by himself, e.g.:

@ABORT-PRINT LINE-PRINTER

If the user appended the current printout, it is aborted and the next

file is taken from the queue.

8.5. Adjusting the Print

Sometimes it is necessary to repeat the printing of the last few pages or to skip the printing of a few pages in the file currently being printed. This is normally done while the spooling is in the stop print mode within the file. When the printing is started the output will continue at the new position. The following two commands are used for this purpose:

8.5.1. @FORWARD-SPACE-PRINT

@FORWARD-SPACE-PRINT <peripheral file name>,<no. of pages>,<no. of lines>

Skip the specified number of pages and lines in the current file, e.g.:

@FORWARD-SPACE-PRINT LINE-PRINTER,10,0

Skip 10 pages in the current file.

8.5.2. @BACKSPACE-PRINT

@BACKSPACE-PRINT <peripheral file name>,<no. of pages>,<no. of lines>

Repeat the printing of the specified number of pages and lines in the current file, e.g.:

@BACKSPACE-PRINT LINE-PRINTER,10,0

Repeat the printing of the last 10 pages.

8.6. Statistics

8.6.1. @LIST-SPOOLING-QUEUE

@LIST-SPOOLING-QUEUE <peripheral file name>,<output file>

List information of a spooling queue on the output device. The listing contains information on the file currently being printed and the files in the spooling queue, e.g.:

@LIST-SPOOLING-QUEUE LINE-PRINTER,,

FILE CURRENTLY BEING PRINTED ON: LINE-PRINTER;;1
(PACK-THREE:USER1)N500-MIC:SYMB;1 , APPENDED BY USER1 , 1 COPY LEFT
APPROX. 283434 BYTES LEFT TO PRINT

```
(PACK-THREE:USER1)N500-MIC-OCTAL:SYMB:1 , APPENDED BY USER1 , 1 COPY
      307602 BYTES IN FILE
LINE-PRINTER:;5 , LAST USED BY ELI-S , 3 COPIES
      51005 BYTES IN FILE
(PACK-THREE:GUEST)JAN:SYMB:1 , APPENDED BY GUEST , 1 COPY
      73 BYTES IN FILE
*** USER MESSAGE: JAN IS READY
(PACK-THREE:GUEST)FILE1:SYMB:1 , APPENDED BY GUEST , 1 COPY
      0 BYTES IN FILE
*** USER MESSAGE: FILE1 IS READY
```

The first three lines show the name of the file currently being printed, showing how many bytes are left to be printed. The remaining lines show the files in the spooling queue. The last two files will cause a user message to be printed before the file is printed. The user message will be printed on the error device.

8.6.2. @SPOOLING-PAGES-LEFT

List the remaining number of pages that can be used by the spooling files. For example:

```
@SPOOLING-PAGES-LEFT
533 SPOOLING PAGES LEFT
@
```

8.6.3. @LIST-SPOOLING-FORM

@LIST-SPOOLING-FORM <peripheral file name>

List the spooling identification key. The key is defined by user SYSTEM (@SET-SPOOLING-FORM) and is compared to the user message of every file to be printed out. The file is only printed if there is a match. Thus, user SYSTEM may select a subset of the files in the spooling queue to be printed.

8.7. Monitor Calls for Spooling

8.7.1. SPCLO (MON 40)

Close a PSF. The call corresponds to @APPEND-SPOOLING-FILE except that the file must be a PSF. The file is put on the spooling queue and a user message can be associated with the file, e.g.:

```
CALL SPCLO(101B,INSERT FORM B',1,1)
```

The PSF 101B is closed and put on the spooling queue. Before it is printed the message 'INSERT FORM B' is printed on the error device and the spooling system waits for a @START-PRINT command.

8.7.2. RSPQE (MON 55)

Read next spooling queue entry and remove it from the queue. This call is meant for user defined spooling systems, e.g.:

LDX	(QENT	% Queue entry destination
LDT	(5	% LDN of spooling device
MON	55	% RSPQE
MON	65	% error return
	...	% normal return

QENT=*

*=#+200

9. Sending Messages to Other Terminals

The SINTRAN III mail system (@MAIL) makes it possible for time-sharing users to send a message to a specific terminal. The full set of subcommands to @MAIL is only available to user SYSTEM. He must initiate the mail system and is the only user permitted to send a message to all terminals by one command(direct broadcast).

Time-sharing users may also send a special message to the operator in order to request some service (@OPERATOR). It is also possible to go into a wait state and come out of it when restarted by the operator (@WAIT-FOR-OPERATOR).

9.1. @MAIL

@MAIL [<output file>]

Enter the mail system. <output file> specifies the destination of the user's own mail, and is only necessary if the user has mail in his mailbox. Time-sharing users can send a message to a specific terminal by using one of two subcommands:

*SEND-MESSAGE <user name>
<message>

The message is terminated by CTRL/L. It is routed to the user's mailbox and he will be notified the next time he logs out or logs on the system. He must then give the @MAIL command in order to read the message.

*SEND-DIRECT-MESSAGE <LDN>
<message>

After typing CTRL/L, the message is routed directly to a specific terminal. It is displayed independently of the terminal activity. (The message will not interfere with the receiver's program execution.)

A list of the subcommands is made available by using the subcommand *HELP. *EXIT returns the terminal to SINTRAN III command mode.

9.2. @OPERATOR

@OPERATOR <text>

Send a message to the error device. The message is terminated by carriage return. (not ' or CTRL/L), e.g.:

(The command is issued on terminal 52)
@OPERATOR MOUNT TAPE 1

On the error device the following message is received:

*** 13.25.15 TERMINAL 52:

MOUNT TAPE 1

9.3. @WAIT-FOR-OPERATOR

Wait for the operator to restart the user (@RESTART-USER). The message received on the error device may look as follows:

---- 13.27.05 WAITING TERMINAL 52

10. NORD-NET

10.1. Introduction

The NORD-NET communication system is an optional part of the SINTRAN III I/O system. Its purpose is to provide communication between two or more independent ND computer systems. The communication can be divided into four categories.

1. Remote terminal communication. A user of a local terminal may use commands and run programs in the remote computer as if his terminal was connected directly to that computer.

2. Remote file access. Files on a remote computer may be accessed by commands or monitor calls as if they were local files. However, then only the functions for open, close, read and write are available

3. Data transfer. A remote and a local program may communicate directly through the channels in a fashion similar to using an internal device.

4. Remote load. The remote computer may be loaded from the local computer. Only main memory can be loaded. Since all communication occurs on serial lines, the line transmission speed may be a limiting factor.

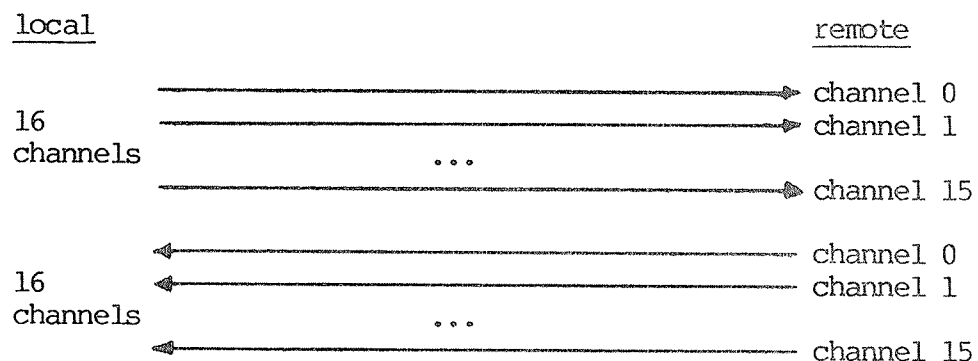


Fig. 16. A Communication Line

This chapter describes points 1 to 3. Remote load is described in the SINTRAN III SYSTEM SUPERVISOR manual. Besides Remote Load, the system supervisor is responsible for starting and stopping the communication on a line (use @START-COMMUNICATION and @STOP-COMMUNICATION).

User RT can associate a password with remote file access (@REMOTE-PASSWORD). A further guide to this command can be found in the manual SINTRAN III REAL TIME GUIDE (ND-60.133).

10.2. The Communication Line

10.2.1. General

The communication line can be divided logically into a maximum of sixteen channels each way (figure 16).

They are numbered from zero to fifteen. If more channels are required, another communication line must be added.

Each channel is provided with a buffer on either side. A buffer is scheduled for transmission either when it is full or when a break character is written to the buffer.

The set of break characters may be chosen by the user.

Information is transmitted in units called communication frames. Acknowledgement for correctly received frames are transmitted together with the frames returned to the sender.

Up to four frames may be transmitted without receiving acknowledgement. This is done by dividing the buffers into four groups.

For each group, the buffer is not discarded until acknowledgement for this group is received. The buffers for sending are always directed to the four groups in a cyclic manner to ensure a correct sequence.

On the receiving side, they are distributed in the same cyclic manner.

When a buffer is transmitted, it is preceded by a buffer header and followed by a cyclic check sum.

A logical device number (LDN) is assigned to the channel on either side. It may be reserved, released, and accessed in a similar manner to any other device in SINTRAN III.

The LDN on either side may be of a different value (figure 17).

The various channels can be interrogated by the commands shown in the next section.

A channel with an associated background program can only be used for remote terminal communication.

Such channels are marked with "BACKGROUND" in the report made by these interrogation commands.

A channel without a background program is used for remote file access and data transfer.

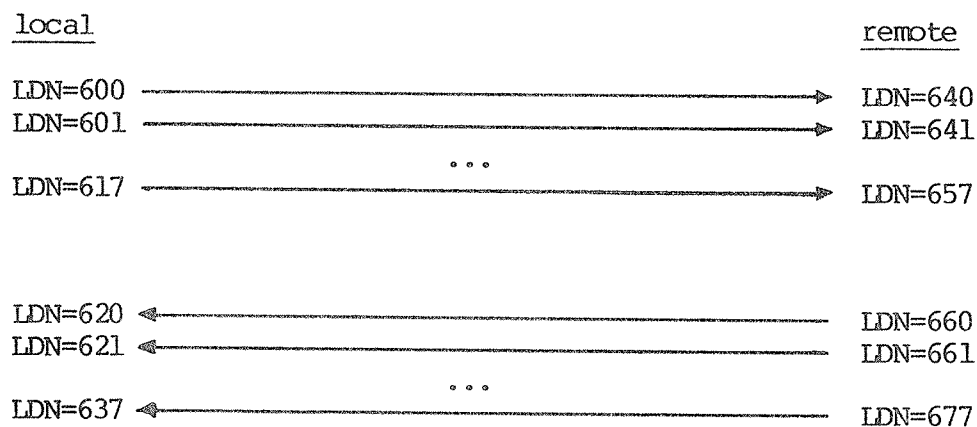


Fig. 17. Example of Logical Device Numbers in NORD-NET

10.2.2. @COMMUNICATION-STATUS

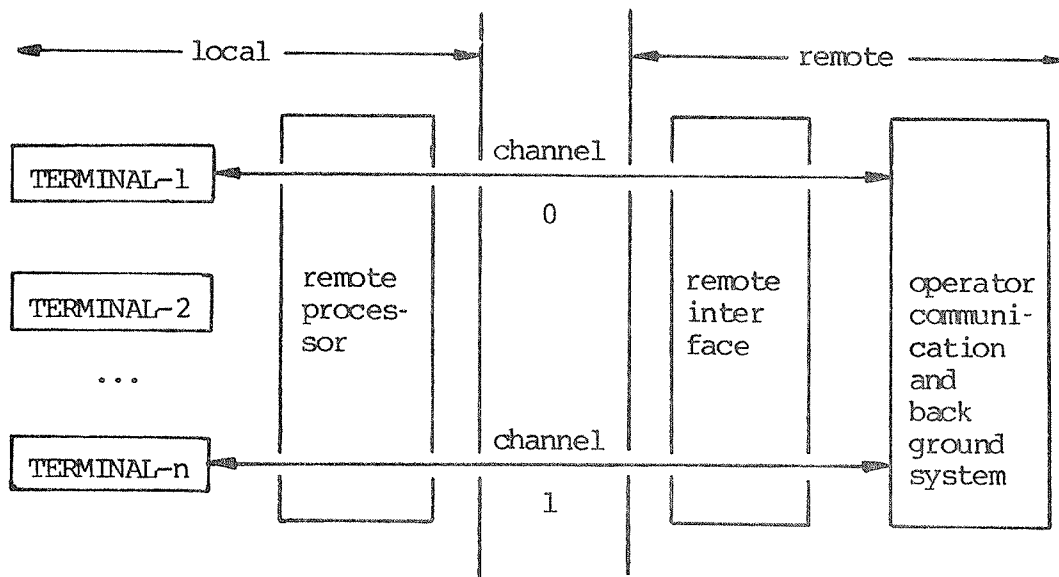
@COMMUNICATION-STATUS <line number>,<output file>

Report the status of the <line number> on the <output file>. The report contains logical device numbers, background vs. data channels, error information and the current communication state.

10.2.3. @COMMUNICATION-LINE-STATUS

@COMMUNICATION-LINE-STATUS <line number>

This command produces an abbreviated report containing only error information and the current communication state.

10.3. Remote Terminals10.3.1. GeneralFig. 18. Remote Terminal Processing

The channels marked BACKGROUND in the report, mentioned above, can be connected to a remote processor. These channels are used for communication between a terminal user in a local SINTRAN III system and the operator's communication and background system in a remote SINTRAN III system. Figure 18 shows the main parts of the NORD-NET implementation.

A terminal user may connect to the remote processor by typing the command @REMOTE <line number> on his terminal. A free channel will be allocated, if available, to the terminal. The user may now LOG IN on the remote system. "rub-out" or "del" puts him temporarily back to the local command processor. The channel is still allocated to the terminal. Another @REMOTE with the same <line number> puts him back to the remote command processor. If he instead types @LOCAL the channel will be disconnected. (A more detailed description is found in section 10.3.5).

For example:

```
...
local processing
...
@REMOTE 1
CHANNEL NUMBERS: LOCAL -600 REMOTE -600
"escape"
15.54.20 18 APRIL 1980
ENTER OLE
PASSWORD:
OK
R@
...
remote processing
...
R@LOGOUT
16.11.34 18 APRIL 1980
-EXIT-
"rub-out"
@
...
local processing
...
```

Remote command mode is indicated by R@ as prompt characters instead of only the @ alone.

A user can be connected to only one line at a time. Thus, if he is connected to remote line 1 and wants to change to remote line 2, it is done as follows,

1. Log out as remote user (R@LOGOUT).
2. Type "rub-out".
3. Type @LOCAL.
4. Type @REMOTE 2
5. Log in as remote user on line 2.

Typing "rub-out" to the remote command processor while in remote command execution mode or remote user mode causes a return to local mode, but the remote processing will continue. The terminal output will be saved and displayed when the user returns to remote command processing.

10.3.2. @REMOTE

@REMOTE <line number>

Connect terminal to remote command processor. If no remote connection exists for this terminal (no @REMOTE since last @LOCAL) a free channel is found and the terminal is connected to the background processor of

the remote computer. If a remote connection already exists, the terminal is connected to this channel. In the latter case, @REMOTE has the reverse function of "rub-out".

10.3.3. @LOCAL

Disconnect remote connection. The communication channel used by the remote connection is released and may be used for other purposes.

10.3.4. Example of @REMOTE and @LOCAL

In this example, @REMOTE and RUB-OUT are used to connect to and disconnect from the remote system.

"escape"

15.25.56 5 SEPTEMBER 1980

VERSION 80.02.01A

ENTER TOM

PASSWORD:

OK

@DATCL

15.26.12 5 SEPTEMBER 1980

@WHO

1 TOM

38 GROUP-4

670 SYSTEM

672 SYSTEM

@REMOTE

CHANNEL NUMBERS: LOCAL -600, REMOTE -600

"escape"

15.25.25 5 SEPTEMBER 1980

ENTER SYS

PASSWORD:

OK

R@DATCL

15.25.39 5 SEPTEMBER 1980

R@WHO

1 RT

670 SYSTEM

672 SYSTEM

384 SYSTEM

R@"rub-out"

VERSION 80.02.01

@DATCL

15.27.03 5 SEPTEMBER 1980

@REMOTE

R@LOG

15.27.01 5 SEPTEMBER 1980

-- EXIT --

"rub-out"

VERSION 80.02.01

@LOCAL

@LOG

15.28.18 5 SEPTEMBER 1980
-- EXIT --

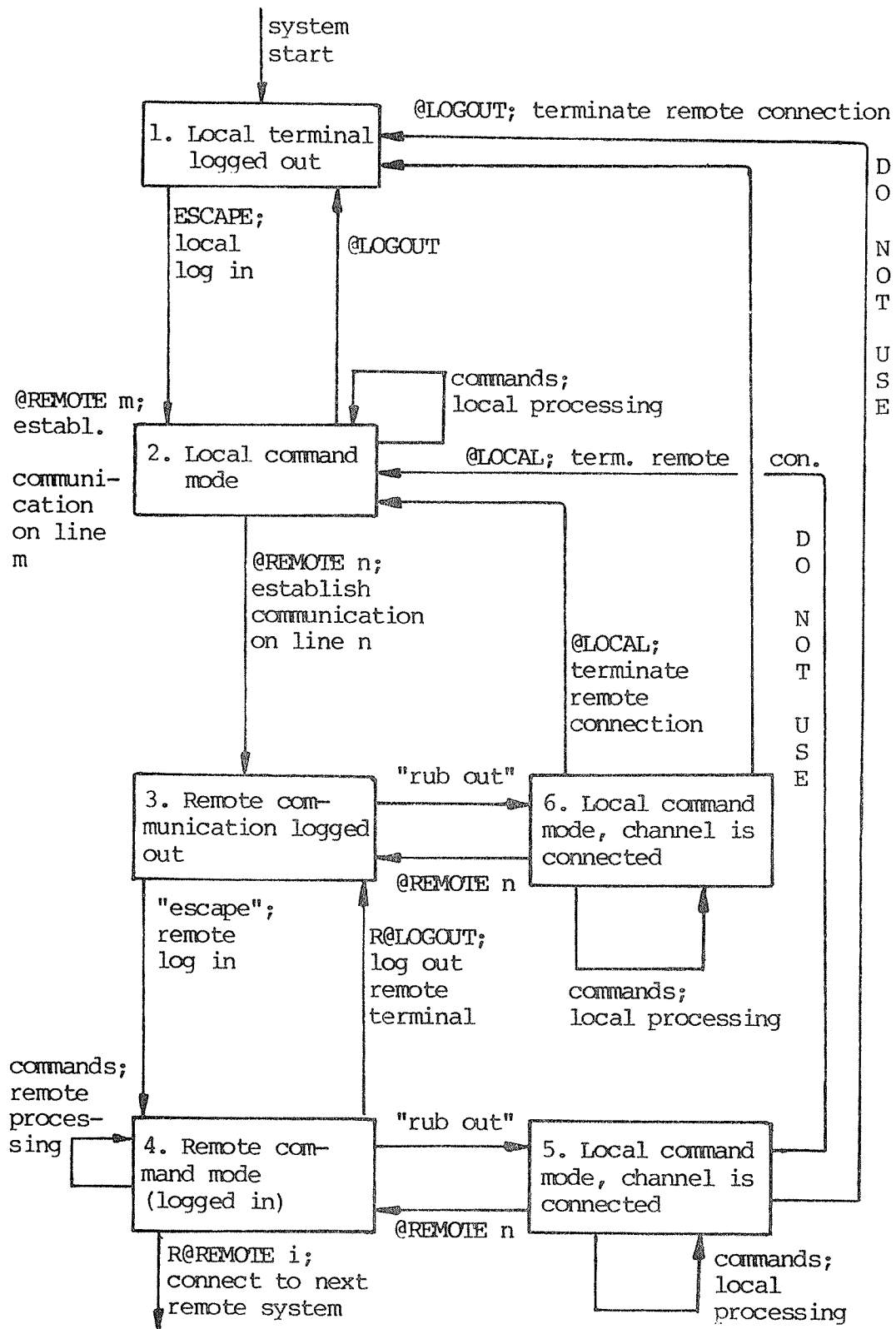


Fig. 19. Remote Processing, State Diagram

10.3.5. Detailed Description of Remote Terminal Connection

This section is a thorough description of the states of remote terminal connection.

The state diagram of remote terminal connection is shown in figure 19. The diagram should be compared to figure 5.

State 1 of figure 5 is equivalent to state 1 of figure 19.

State 2 of figure 5 is equivalent to the other states of figure 19 except the response to @REMOTE.

This command establishes communication by reserving a free channel ,if any. The state is changed to 3.

Normally, the user proceeds to state 4 by logging on remotely.

All commands will now be processed on the remote system. Typing "rub-out" causes a transfer to local command mode, state 5.

Note that "rub-out" may be entered while in remote command processing mode or remote user mode.

This causes any remote processing to proceed as an independent process while the next commands are processed locally.

Any remote terminal output during state 5 will be collected and displayed when returning to state 4.

Another @REMOTE n command will cause a transfer to remote command mode, state 4, using the same channel as before.

In state 5 it is possible to terminate the remote connection (@LOCAL or @LOGOUT), but this should be avoided. The remote processing may be left in an indeterminate state. Instead, go back to state 3, type "rub-out" to change to local communication, and type @LOCAL to terminate the remote connection. (@LOGOUT will also terminate the connection.)

State 6 is equivalent to state 5 with respect to handling @REMOTE n. The state is changed to remote communication, state 3, using the same channel as before. Nesting of remote connections is performed in state 4 by typing another @REMOTE m command. It could even be a remote connection back to the local system.

This is necessary for the type of processing shown in figure 20. Here, the user runs a remote program which uses one or more files in the local system.

The communication starts with the first @REMOTE n command, establishing the interactive dialog on channel a. The user must then establish channel b by means of a @REMOTE m command back to the local system.

Then log onto the local system and finally type "rub-out" to get back to the remote system. The user may now start the remote program (program x) which can be run either in foreground or background.

The program will use a third channel for data transfer (channel c) while channel b will be used for administration.

Transfer of data can only occur directly between two systems which have a direct connection.

If system A connects to system B and B connects to system C, it is possible to connect to B, log in on B, connect to C and log on to C. However, any data transfer from A to C must first be made to B and then to C.

An intermediate file or program in B will take care of this problem.

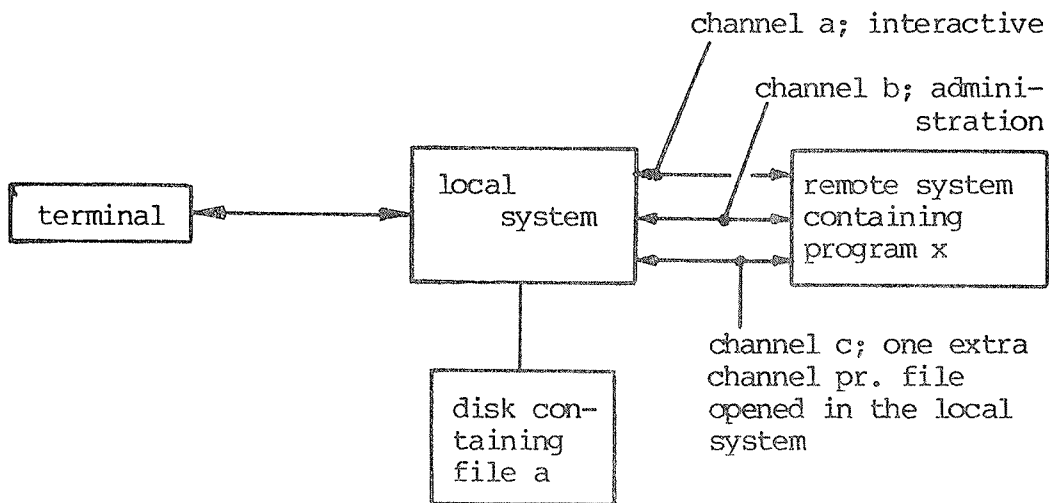


Fig. 20. Remote Program Using a Local File.

10.4. Remote File Access

Files on a remote ND system may be opened and closed from the local system. Only open, close, input and output functions are permitted on remote files. The communication channel to be used is specified as a prefix to the file name. To open a remote file from a background program, the user must be logged in on both systems. To open a remote file from a foreground program, the password of user RT on the remote system must be set by the @REMOTE-PASSWORD command. In the following example, the local file LFIL is read by QED and written to the remote file RFIL. The channel to be used is CHANNEL-1 and must have been defined as a peripheral file by user SYSTEM (@SET-PERIPHERAL-FILE). LFIL is owned by user PER and RFIL by user OLE.

The use of CHANNEL-1 as a prefix to the remote file name is explained at the end of this section.

```

"escape"
12.1.06 28 AUGUST 1980
ENTER PER
PASSWORD:
OK
@REMOTE
CHANNEL NUMBERS: LOCAL - 600, REMOTE - 600
"escape"
12.10.12 28 AUGUST 1980
ENTER OLE
PASSWORD:
OK
R@"rub-out"
@OED
OED 3.8
*R LFIL
2480 WORDS READ
*W CHANNEL-1.RFIL
2480 WORDS WRITTEN
*EX
@REMOTE
R@LOG
12.12.04 28 AUGUST 1980
-- EXIT --
"rub-out"
@LOG
12.12.15 18 AUGUST 1980
-- EXIT --

```

Note that when the remote file is accessed, the own user name is the name used when logging in on the remote system. The file RFIL is expected to be among the files owned by user OLE or user SYSTEM on the remote system.

When processing the command *W CHANNEL-1.RFIL, two channels are used. The first is the one allocated to commands. The second is the channel corresponding to the peripheral file name CHANNEL-1.

The next example is a compilation in the local system. The FORTRAN source file LFIL is compiled. The compiler listing is output to the remote line-printer and the object code is output to the new remote file OBJ:BRF belonging to user OLE.

```

"escape"
12.25.18 28 AUGUST 1980
ENTER PER
PASSWORD:
OK
@REMOTE
CHANNEL NUMBERS: LOCAL - 600, REMOTE - 600
"escape"

```

```
12.25.30 28 AUGUST 1980
ENTER OLE
PASSWORD:
OK
R@"rub-out"
@FTN
NORD-10 FTN COMPILE
$COM LFIL,CHANNEL-1.L-P,CHANNEL-2."OBJ"
189 STATEMENTS COMILED, OCTAL SIZE = 3122
CPU-TIME USED IS 6.9 SEC.
$EX
@REMOTE
R@LOG
12.32.31 28 AUGUST 1980
-- EXIT --
"rub-out"
@LOG
12.32.40 28 AUGUST 1980
-- EXIT --
```

In general, the syntax of a remote file name is:

<channel name>.<local file name>

Only one level of remote connection can be specified. For ex.:

```
CHANNEL-1.(PACK-ONE:PER)FILA:SYMB;2
CH-2."FILE-2"
KANAL4.(SYS)FTNLIBR:BRF
```

The file number returned from a remote @OPEN-FILE, OPEN statement or OPEN monitor call will be the logical device number of the channel as defined in the local computer. After the file is opened, a remote file may be accessed by READ and WRITE statements, INBT and OUTBT monitor calls, etc.

10.5. Data Transfer

10.5.1. General

Any free channel not dedicated to background programs may be used for data transfer. The channel will then have a function similar to an internal device. The only difference is that the sending and the receiving programs are in two different systems. The following rules apply:

1. The channel must be reserved by the user programs. The logical device number on either side is used for reservation.

2. The receiving program asks for input (by using a monitor call or statement) and is set in a waiting state until data is received through the channel.

3. The sending program outputs to the channel (by using a monitor call or statement) and is set in a waiting state under one of the following conditions:

- the receiving program has not asked for input,
- a break character is sent,
- the buffers available are almost full, or
- a wait acknowledge (WACK) is received from the channel.

The sending program is restarted when a request for input is received through the channel. The request is sent from the receiving program when it detects a break character.

A wait acknowledge is sent if the input queue for a channel exceeds a predefined number of buffers. (The number is defined at system generation time.) The wait will prevent one channel occupying the whole buffer pool if the receiver reads data at a lower rate than the sending program.

A wait acknowledge simulates a break character at the end of the last transmitted buffer on the channel.

4. The break strategy may be defined by the receiving program. The strategy is transmitted to the sending system as a special buffer. It is possible to specify that no break is permitted.

A buffer will then be transmitted only if it is full or the sending program executes CLOSE-FILE or IOSET function on the channel. The break strategy should cause as few breaks as possible in order to reduce the system overhead.

WFILE - Write a block of data to a communication channel.

In the following example, data is received from the communication channel having the LDN = 600:

```

      LDA (REPAR      %
MON   122            % RESERVE CHANNEL
      LDT (600        %
MON   13             % CLEAR INPUT BUFFER
      JMP ERROR       % ERROR EXIT
      SAA -1          %
MON   4              % SET BREAK STRATEGY
LOOP, LDT (600        %
MON   1              % INPUT A BYTE
      JMP TEST        % TEST FOR ERROR
      .
      .
      .
      JMP LOOP        %
      .
      .
      .
% IF ERROR = 161, NO ANSWER FROM DEVICE,
% THEN TRY AGAIN AFTER 5 SEC. (CAN BE LIMITED
% TO FOR EX. 40 RETRIES.) THE DRIVER HAS NO
% RETRY FACILITY.
TEST, SAT 161        % T = 161
      SKP EQL SA DT   % TEST FOR 161
      JMP ERROR       % NOT 161 - ERROR
      LDA (HPAR       WAIT 5 SEC.
MON   104            % HOLD
      JMP LOOP        % TRY AGAIN
      .
      .
      .
REPAR, (600          % CHANNEL NUMBER
      (0             % INPUT BUFFER
      (0             % WAIT FOR RESERVATION
HPAR,  (5             % 5 SEC.
      (2             % UNIT OF SECONDS
)FILL

```

The input buffer should always be cleared, since the previous program using the channel may have been terminated abnormally.

The following FORTRAN program will write a record to the channel:

```
      ...  
      I=RESRV(600B,1,0)  
      I=IOSET(600B,1,0,-1)  
      ...  
      WRITE(600B,10) ...  
10    FORMAT( ...  
      ...
```

The corresponding program to read is:

```
      ...  
      I=RESRV(600B,0,0)  
      ...  
      READ(600B,10) ...  
10    FORMAT( ...  
      ...
```

10.5.2. WRQI (MON 163)

Place the calling program in a wait state until a request for input is received from the remote system. The call is useful in interactive communication programs when the local echoing should wait until the receiving program asks for input.

A P P E N D I X A

Editing Control Characters for Commands

The following list contains the subset of QED control characters which are relevant for SINTRAN III commands.

CORRECTING THE CURRENT LINE

CTRL/A Backspace one character (echo is up-arrow or under-line)

CTRL/Q Restart the command on a new line (echo is left-arrow)

COPY FROM PREVIOUS LINE

CTRL/C Copy one character

CTRL/D Copy the rest of the line including carriage return

CTRL/H Copy the rest of the line not including carriage return

CTRL/Zx Copy the old command up to and including x

CTRL/Ox Copy the old command up to but not including x

SKIP CHARACTERS IN OLD COMMAND

CTRL/S Skip one character

CTRL/Xx Skip characters up to and including x

CTRL/Px Skip characters up to but not including x

(Carriage return skips the rest of the line.)

INSERT CHARACTERS

CTRL/E Change to insert mode (echo is <) or overwrite mode (echo is >)

A P P E N D I X B

Index

This index includes terms which are not complete headings. For names of time-sharing commands and monitor calls, the reader should make himself familiar with the table of contents.

Some commands and monitor calls are only referenced in this manual and are therefore included in the index.

* indicates a definition of the term. It occurs only in a list of multiple references.

access
 append 3.5.1
 common 3.5.1
 default 3.5.1, 3.10.7, 3.5.4*
 explicit 3.6
 friend 3.5.4, 3.10.7, 3.5.1*
 implicit 3.6
 "normal default" 3.5.4, 3.5.1*
 own..... 3.5.4, 3.10.7, 3.5.1*
 public..... 3.5.4, 3.10.7, 3.5.1*
 random..... 3.8.6, 3.2.1*
 sequential..... 3.2.1
accounting system..... 2.7
ACM (MON 145)..... 1.3
active batch state
 see batch
attention character
 see character

background
 program..... 1.1
backup..... 3.9, 3.11, 3.13
basic time unit
 see time
batch
 active state..... 7.1
 idle state..... 7.1
 job..... 7.6
 output device..... 2.5
 passive state..... 7.1
 processor..... 7.1
 program..... 2.10.3
 program termination..... 5.7.3
 queue..... 7.2
 time used in job..... 4.3.5
 see also error
binary program
 see program
binary relocatable file..... 3.2.6, 3.2.3*
block..... 3.1, 3.2.1
 beginning of..... 3.8.6
 size..... 3.8.4, 3.8.6, 3.2.1*
BPUN command..... 5.5

BPUN format see program
 break strategy..... 3.8.3.7, 6.2.8*
 BRF see,
 binary relocatable file
 broadcast..... 9.1
 buffer
 input..... 3.8.6.10
 output..... 3.8.6.11
 byte..... 3.2.1
 maximum b. pointer..... 3.8.6
 pointer..... 3.8.6

 cabeling..... 2.2
 call, standard format..... 2.10.3
 character
 attention..... 7.3
 control..... 2.4.6, 3.2.9, 5.8
 device..... 3.2.7
 in files..... 3.2.1
 string..... 2.4.1, 3.8.3.8
 command
 element..... 2.4.1
 execution mode..... 2.3
 mode..... 5.3, 2.3*
 name..... 4.2.3
 parameter..... 2.4.1
 in programs..... 5.7.2
 subfield..... 2.4.1
 system supervisor..... 1.3
 time-sharing..... 1.3
 see also error,
 real-time
 comment card..... 7.6.2
 communication mode..... 6.2.4, 6.2.1*
 configuration
 minimum hardware..... 1.4
 software..... 1.1
 connect time..... 2.6
 console terminal
 see terminal
 contiguous see file
 control character
 see character

 data link..... 3.8.3.1
 data storage medium..... 3.1
 date..... 4.3
 default directory
 see directory
 delay on CARRIAGE RETURN..... 6.2.1
 device..... 3.1, 3.7.3, 3.7.4
 element..... 3.2.7
 mass storage..... 3.7
 reserve..... 7.6

- word oriented..... 3.8.3.1
 - see also batch,
 - character, error
- @DEVICE-FUNCTION..... 3.11, 1.3*
- direct program entry
 - see MAC
- directory
 - default..... 2.6, 3.2.2, 3.3.1*
 - main..... 3.3.1
 - on mass storage medium..... 3.1
 - see also file
- duplex, full..... 2.9
- dynamic LDN
 - see logical device number
- echo strategy..... 6.2.8*
- edited input..... 3.2.9
- element see command
- end-user..... 1.2
- error
 - in batch jobs..... 7.6
 - command processor..... 2.5
 - device..... 2.5, 7.6.8.1, 8.2.9.2, 2.1*
 - file system..... 2.5
 - FORTRAN run time..... 2.5
 - run-time..... 2.5
 - system..... 2.1, 6.1, 2.5*
- element see command
- ESCAPE (ESC)..... 6.2.2, 6.2.3, 6.2.5, 6.2.6
- execution
 - mode..... 7.7.1
 - see also command
- explicit creation
 - see file
- extent..... 3.2.1*
- file
 - contiguous..... 3.4.1, 3.6, 3.2.1*
 - directory name..... 3.2.2
 - explicit creation..... 3.4
 - implicit creation..... 3.4
 - indexed..... 3.2.6, 3.8.4, 3.4
 - label..... 3.10.9
 - mass storage..... 3.2.1, 3.1*
 - name..... 2.4.1
 - object name..... 3.2.2
 - object number..... 3.10.3
 - owner..... 3.1
 - owner name..... 3.2.2
 - peripheral..... 3.7
 - peripheral name..... 3.1

- peripheral spooling..... 8.1
- permanently opened..... 2.10.4, 3.2.6, 3.8.2.4*
- scratch..... 3.8.2.3
- segment..... 4.3.4
- spooling..... 3.4.9
- temporary..... 3.4.9
- type..... 3.2.2
- see also binary
- relocatable file,
- character, error,
- job
- floppy disk..... 3.3.1, 3.3.2, 3.11
- foreground program
- see program
- formatting of disk
- see "virgin" disk
- FORTRAN i/o..... 3.2.1
- FORTRAN run-time error
- see error
- FORTRAN unit number
- see unit number
- full duplex see duplex

- GRAPHIC (MON 155)..... 1.3

- hardware configuration
- registers..... 2.10.3
- see also configuration
- HDLC (MON 201)..... 1.3
- HOLD see no-wait mode
- hole..... 3.2.1

- identification key
- see spooling
- idle batch state
- see batch
- implicit creation
- see file
- indexed file see file
- insert mode..... 2.4, appendix A
- interrupt level..... 2.10.3
- @IOSET..... 1.2
- IOSET (MON 141)..... 1.3

- job
- input file..... 7.2
- output file..... 7.2
- see also batch, mode
- remote job entry

label see file or reel
LDN see logical device number
@LIST-DEVICE FUNCTION..... 1.3
@LIST-REMOTE-QUEUE..... 1.3
@LOCAL..... 1.3
logical device number..... 2.6, 3.1, 3.2.7*
 dynamic..... 3.10.3, 3.2.7*
 static..... 3.7, 3.7.5, 3.7.6, 3.2.7*
logical page see page

MAC..... 2.10.2, 3.2.7
 direct program entry..... 2.10.2
machine code..... 1.3
MAGTP (MON 144)..... 1.3
mail box..... 9.1
main directory
 see directory
mass storage device
 see device
mass storage file
 see file
mass storage medium
 see directory
memory
 alternative virtual..... 5.4.2
 virtual..... 5.4.2
missing carrier..... 6.2.1
mode
 job..... 7.2
 nesting..... 7.5.1
modem..... 2.9.2
monitor call..... 2.10.1, 2.10.2, 2.10.3, 1.3*

ND-number..... 6.1
nesting see mode
no-wait mode..... 3.8.3.7
 HOLD in..... 3.8.3.7
NORD-50..... 1.3
ND Relocating Loader..... 2.10.1, 3.2.6
NRL see ND Relocating Loader
number
 printing of..... 3.8.3.6
numeric parameter..... 2.4.1

object entry see file
object name see file
"old line" editing..... 5.7
overstrike..... 2.4, appendix A

"own" terminal
 see terminal
 owner name see file

page..... 3.2.1
 bad p. on disk..... 3.11
 logical p..... 4.3.4
 paper crash..... 8.1
 passive batch state
 see batch
 password..... 1.2.1, 1.2.2, 2.2, 2.8*
 peripheral
 standard..... 1.4
 see also file
 priority..... 1.2.1
 PROG-file see program
 program
 binary (BPUN)..... 5.5, 5.1*
 foreground 3.6, 1.1*
 PROG-file..... 5.1, 5.3.1, 3.2.3*
 restart..... 5.3.2
 start..... 5.3.1
 terminating in..... 5.7, 5.9
 see also background,
 batch, command
 PSF see peripheral spooling
 file
 PT see page

random access see access
 real-time (RT)
 command..... 1.3
 description..... 4.3.3
 description address..... 4.3.5
 program..... 1.2.2
 programming..... 1.1
 see also
 user record..... 3.1, 3.2.1*
 reel label..... 3.10.9
 reentrant subsystem
 see subsystem
 registers see hardware
 @REMOTE..... 1.3
 remote job entry (RJE)..... 7.1
 reserve device
 see device
 restart address..... 2.10.4
 REWIND statement..... 5.7
 RJE see remote job entry
 RT see real-time
 run-time see error

- scratch file
 - see file
- segment..... 4.3.4
 - see also file
- selection..... 2.4.1
- sequential access
 - see access
- SINTRAN III/RT..... 1.1
- SINTRAN III/VS..... 1.1
- software configuration
 - see configuration
- source code see symbolic code
- spooling
 - identification key..... 8.6.3
- system states..... 8.1
 - see also file
- standard call format
 - see call
- standardization of terminal
 - see terminal
- static LDN
 - see logical device
 - number
- stop condition..... 8.1
- subfield see command
- subsystem..... 2.5, 2.3*
 - reentrant..... 5.1, 4.2.1
- SYMB-file..... 5.1, 3.2.3*
 - see also symbolic code
- symbolic code..... 2.10.1
 - see also SYMB-file
- system
 - management..... 1.2.3
 - identification..... 4.2.2
 - see also error
- system supervisor (user SYSTEM)
 - see command, user
- telephone lines..... 2.9
- temporary data..... 3.2.6
 - see also file
- terminal
 - communication characteristics..... 6.2, 6.1*
 - console..... 2.5, 3.2.9, 2.1*
 - standard..... 2.1
 - standardization of..... 6.1, 3.2.5*
 - type name..... 6.1, 3.2.5*
 - type number..... 3.2.5
 - wait state..... 6.4
- termination
 - see batch, program,
 - user
- time
 - basic t. unit..... 4.2.

CPU..... 4.3.5
 of day..... 2.6
 internal..... 4.3.4
 measurement..... 2.6
 queue..... 4.3.2
see also batch,
 connect
 time-sharing
 see command, user
 TRACB (MON 156)..... 1.3
 transmission speed..... 2.2
 TSS operating system..... 3.8.4.4
 type see file

unit number..... 3.1
 upper case..... 6.2.1
 user
 area..... 3.1
 mode..... 5.3, 2.3*
 name..... 1.2.1, 2.2, 3.1*
 number..... 7.7.1
 terminal..... 2.1
 time-sharing..... 1.2
 RT..... 1.2
 SYSTEM..... 1.2.3

version..... 3.2.2
 "virgin" disk, formatting of..... 3.11
 virtual memory
 see memory

wait mode..... 3.8.3.7
 see also terminal
 WRQI (MON 163)..... 1.3

XMSG (MON 200)..... 1.3

SEND US YOUR COMMENTS!!!

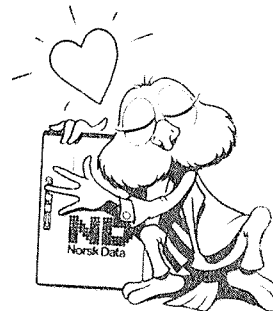


Are you frustrated because of unclear information in this manual? Do you have trouble finding things? Why don't you join the Reader's Club and send us a note? You will receive a membership card — and an answer to your comments.

Please let us know if you

- find errors
- cannot understand information
- cannot find information
- find needless information

Do you think we could improve the manual by rearranging the contents? You could also tell us if you like the manual!



HELP YOURSELF BY HELPING US!!

Manual name: SINTRAN III Timesharing/Batch Guide

Manual number: ND-60.132.03

What problems do you have? (use extra pages if needed) _____

Do you have suggestions for improving this manual ? _____

Your name: _____ Date: _____

Company: _____ Position: _____

Address: _____

What are you using this manual for ? _____

NOTE!

This form is primarily for documentation errors. Software and system errors should be reported on Customer System Reports.

Send to:

Norsk Data A.S
Graphic Center
P.O. Box 25, Bogerud
0621 Oslo 6, Norway

Norsk Data's answer will be found on reverse side



Answer from Norsk Data

Answered by

Date

Norsk Data A.S
Graphic Center
P.O. Box 25, Bogerud
0621 Oslo 6, Norway

Systems that put people first

NORSK DATA A.S OLAF HELSETS VEI 5 P.O. BOX 25 BOGERUD 0621 OSLO 6 NORWAY
TEL.: 02 - 29 54 00 - TELEX: 18284 NDN