Reference Manual

RDOS COMMAND LINE INTERPRETER

093-000109-00

Ordering No. 093-000109

Printed in the United States of America
Rev. 00, February 1975

NOTICE

(RDOS 04) Original Release - February 1975

PREFACE

This manual discusses the Command Line Interpreter (CLI) which is the primary interface between the user at a system console and the Real Time Disk Operating System (RDOS). Included in the text are discussions on the following subjects:

Chapter 1 - Introduces the CLI, discusses the operating environment and summarizes the various CLI commands.

Chapter 2 - Discusses the CLI operating considerations.

Chapter 3 - Discusses all the CLI commands which are divided into functional groupings.

Appendix A - Summarizes the CLI syntax and run time errors.

Appendix B - Discusses the format of dump files.

Appendix C - Discusses keyboard command interpretation.

It is assumed that the user of this manual is familiar with the programs that may be requested by the use of CLI, but for reference purposes the following applicable documentation summary is included.

017-000016	ALGOL Cache Memory Management
093-000018	Text Editor's User's Manual
093-000040	Extended Assembler
093-000041	Relocatable Math Library File
093-000044	The Symbolic Debugger User's Manual
093-000052	Extended ALGOL User's Manual
093-000053	Fortran IV User's Manual
093-000065	Extended Basic User's Manual
093-000068	Fortran IV Run Time Library User's Manual
093-000074	Library File Editor
093-000075	Real Time Disk Operating System User's Manual
093-000080	Extended Relocatable Loader User's Manual
093-000081	Macro Assembler User's Manual
093-000083	Introduction to the Real Time Disk Operating System
093-000084	Octal Editor
093-000085	Fortran 5 User's Manual
093-000087	Batch User's Manual
093-000095	Real Time Input/Output System User's Manual
093-000096	Fortran 5 Run Time Library User's Manual
093-000105	RDOS User's Handbook.

093-000106	Summary of Available Software
093-000107	Fortran Commercial Subroutine Package
093-000108	NOVA® * Sort Merge
093-000110	NOVA® Software Summary and Bibliography
093-000111	Superedit Reference Manual (For the NOVA and ECLIPSE™ * line computers.)

*ECLIPSE is a trademark and NOVA is a registered trademark, of Data General Corporation, Southboro, Massachusetts.

TABLE OF CONTENTS

CHAPTER 3 - CLI COMMANDS

CHAPTER 3 - CLI COMMANDS (Continued)

CHAPTER 1

CONSOLE USER'S GUIDE

INTRODUCTION

The Command Line Interpreter (CLI) is the primary interface between a user, at the system console and the RDOS operating system. (The system console can either be the Teletype* or a video display console.)

CLI is a system program designed to accept command lines, manually entered by a user from the system console. Command lines are translated and are recognized as commands by the operating system.

The CLI program is easy to use, that is, command lines are made up of alphanumeric characters and special characters entered according to certain syntax rules.

Specific error messages are reported on the system console which indicate errors, as they occur, for user convenience.

FOREGROUND/BACKGROUND USES

Memory can be divided into two distinct areas, foreground and background. Foreground tasks may have priority over background tasks or they may compete for resources on an equal priority basis.

The CLI can operate in either foreground or background.

The CLI also has the capability of modifying the size of both foreground and background memory size by executing the command SMEM.

PROGRAM REQUEST

The CLI requests a user at the system console to specify a command line of valid input, entered according to syntatical rules, ended by an acceptable terminator: either a carriage return ( ), or simultaneous depression of CTRL and L keys.

*Teletype is a registered trademark of Teletype Corporation, Skokie, Illinois.

PROGRAM REQUEST (Continued)

It should be noted that both line terminators are interchangeable and that the keyboards associated with the teletype and video display console are functionally the same.

PROGRAM DEVELOPMENT

All of the programs associated with program development are requested for execution by user entered CLI commands. Programs associated with program development include the following:

- Editors - Text, Octal, multi-terminal, library, SPEED
- Assemblers - Extended, Macro
- Compilers - Fortran IV, Fortran 5, Algol
- Loaders - Relocatable, Overlay

Additionally, the CLI is used: to load user programs once program development is complete, and activate the BATCH monitor (BATCH permits user jobs to be executed serially without the need for continuous operator intervention, since job control commands are imbeded as part of the job stream. Additional information regarding BATCH is available in the BATCH User's Manual (093-000087)).

FILE MAINTENANCE

The term file applies to any collection of information or to any device receiving or providing the information. Typical examples of both file types are:

- source program file
- relocatable binary file
- listing file
- core image file (save file)
- teletype keyboard
- cassette unit, etc.

The CLI provides the means of performing file maintenance by its associated commands. The following are typical examples of file maintenance:

- create a file
- delete a file
- rename a file
- concatenate two or more files
- set file use count to zero
- change link access attributes, etc.

ADDITIONAL CAPABILITIES

The CLI program includes commands allowing the following:

- setting and optaining the time of day
- naming the current operating system
- enable, disable and stop a spooling operation
- overwrite the CLI with a program chain
- requesting the SYSGEN program

INTERRUPTING CLI

The action of CLI can be interrupted by a user at a system console by entering certain reserved key combinations. These program breaks cause the following actions to occur:

- terminate the current CLI command

- terminate a foreground program

SUMMARY

This manual includes all commands applicable to CLI but does not address itself to the operating characteristics of the programs callable by CLI. The reader is directed to the applicable documents (i.e., SYSGEN discussed in RDOS User's Manual, etc.)

Table 1-1 provides a summary of all the applicable CLI commands. Each of the commands are discussed in detail in Chapter 3 of this manual.

Table 1-1. CLI Commands

COMMANDS THAT INVOKE UTILITIES

ALGOL	-	Compile an ALGOL source file.
ASM	-	Assemble a source file.
BASIC	-	Execute a BASIC program.
BATCH	-	Execute a BATCH job stream.
CLG	-	Compile load and execute a FORTRAN IV program
DEB	-	Load a program into memory and go to the debugger.
EDIT	-	Edit an ASCII file.
FORT	-	Compile and assemble a FORTRAN IV program.
FORTRAN	-	Compile and assemble a FORTRAN 5 program.
LFE	-	Update library file.
MAC	-	Perform a macro assembly.
MEDIT	-	Multi-Editor
OEDIT	-	Edit a disk file.
OVLDR	-	Create an overlay replacement file.
RDOSSORT	-	RDOS Sort Merge.
REPLACE	-	Replace overlays in overlay file.
RLDR	-	Perform a relocatable load.
SPEED	-	Edit an ASCII file.

FILE MAINTENANCE COMMANDS

File Creation

CCONT	-	Create a contiguous file.
CRAND	-	Create a random file.
CREATE	-	Create a sequential file.

File Handling

APPEND	-	Concatenate two or more files.
BPUNCH	-	Punch a binary file to the high speed punch ($PTP)

Table 1-1. CLI Commands (Continued)

FILE MAINTENANCE COMMANDS (Continued)

File Handling (Continued)

BUILD	-	Build a file containing file names.
CHATR	-	Change a file's attributes.
CHLAT	-	Change link access attributes.
CLEAR	-	Set file use count to zero.
DELETE	-	Delete a file or series of files.
DUMP	-	Dump one or more files.
ENDLOG	-	Close the log file (stop recording console output).
FILCOM	-	Compare two files.
FPRINT	-	Print a disk file in bytes, decimal, hexedecimal or octal.
LOAD	-	Reload dumped files.
LOG	-	Open the log file (Record all CLI dialogue).
MKABS	-	Make an absolute file from a core image file.
MKSAVE	-	Make a save file from an absolute binary file.
MOVE	-	Move files from one directory to another.
PRINT	-	Print a file on the line printer.
PUNCH	-	Copy an ASCII file on the high speed paper tape punch.
RELEASE	-	Release a device or directory from the system.
RENAME	-	Change the file name.
REV	-	To display the revision level of a save file.
SAVE	-	Save a core image as a save file.
TYPE	-	Output file contents on the system console.
UNLINK	-	Delete a link entry name.
XFER	-	Copy the contents of a file into another file.

Directory Commands

CDIR	-	Create a subdirectory.

Table 1-1. CLI Commands (Continued)

FILE MAINTENANCE COMMANDS (Continued)		
Directory Commands (Continued)		
CPART	-	Create a secondary partition.
DIR	-	Change the current default directory.
EQUIV	-	Assign a new name to a directory specifier.
GDIR	-	Print the current device directory name.
INIT	-	Initialize a directory or device.
LINK	-	Create a link to a file on the same or another directory.
LIST	-	List file directory information.
MDIR	-	Get the name of the current master directory.
FOREGROUND-BACKGROUND COMMANDS		
EXFG	-	Execute in the foreground.
GMEM	-	Get the foreground/background memory size in a mapped system.
SMEM	-	Set the foreground/background memory size in a mapped system.
SYSTEM GENERATION COMMANDS		
SYSGEN	-	Generate an operating system.
SQUASH	-	Prepare a system save file for use in bootstrapping.
TUNING FILE COMMANDS		
TPRINT	-	Print the tuning file.
TUOFF	-	Turn off the system tuning facility.
TUON	-	Turn on the system tuning facility.

Table 1-1. CLI Commands (Continued)

SPOOLING COMMANDS		
SPDIS	-	Disable device spooling
SPEBL	-	Enable device spooling
SPKILL	-	Stop a spooling operation
TIME, DATE AND SYSTEM NAME COMMANDS		
GSYS	-	Get the name of the current operating system.
GTOD	-	Get the time and the date.
SDAY	-	Set today's date.
STOD	-	Set the time.
BOOTSTRAP COMMAND		
BOOT	-	Perform a disk bootstrap.
CHAIN COMMAND		
CHAIN	-	Overwrite the CLI with a program chain.
DISK BLOCKS AVAILABLE/USED COMMAND		
DISK	-	List the number of disk blocks used and remaining.
COMMUNICATION COMMAND		
MCABOOT	-	Transmit a program via a MCA line.
HIGHER LEVEL PROGRAM COMMAND		
POP	-	To return to the next higher program in this program environment.

CHAPTER 2

OPERATING CLI

INTRODUCTION

This chapter explains the usage considerations associated with the Command Line Interpreter (CLI). Included are discussions on the following subjects:

- Standard character set
- CLI symbols
- File information (general)
- CLI conventions
- CLI searches
- System console breaks
- CLI error messages

CHARACTER SET

The standard ASCII character set (shown in Table 2-1) is available to the CLI user at the system console.

SPECIAL SYMBOLS

A few ASCII characters have special syntatical significances. These characters and their meanings are summarized in Table 2-2, and are discussed in detail in the following text.

GENERAL FILE INFORMATION

A file is a logical collection of information, or a physical device transmitting or receiving information.

A core image file is stored in disk, word for word, exactly as it will be loaded into memory starting at location 16_8.

All device and disk files are accessible by file name; all cassettes and magnetic tapes are accessible by a file number.

Table 2-1 Standard ASCII Character Set

7-Bit Octal Code	Character	7-Bit Octal Code	Character	7-Bit Octal Code	Character	7-Bit Octal Code	Character
000	Null*	072	:	125	U	165	u
011	Tab	073	;	126	V	166	v
012	LF*	074	<	127	W	167	w
014	FF	075	=	130	X	170	x
015	CR	076	>	131	Y	171	y
040	SP	100	@	132	Z	172	z
041	!	101	A	141	a	177	Rubout*
042	"	102	B	142	b		
043	#	103	C	143	c		
046	&	104	D	144	d		
052	*	105	E	145	e		
053	+	106	F	146	f		
054	,	107	G	147	g		
055	-	110	H	150	h		
056	.	111	I	151	i		
057	/	112	J	152	j		
060	0	113	K	153	k		
061	1	114	L	154	l		
062	2	115	M	155	m		
063	3	116	N	156	n		
064	4	117	O	157	o		
065	5	120	P	160	p		
066	6	121	Q	161	q		
067	7	122	R	162	r		
070	8	123	S	163	s		
071	9	124	T	164	t		

Table 2-2 Symbols and Conventions Used in Command Line Syntax

Symbol	Usage	Example
) or ↓	) (carriage return key) terminates an input command line and activates the CLI. (Control and L key) is identical to the carriage return.	CREATE A B) CREATE A B ↓
\	\ (shift and L keys). Delete an entire line.	CCREAGE \ )
←	← (RUBOUT key). Erase the last entered character before the ←.	CC←READ←TE
, (space)	Arguments are separated by commas or spaces. Extra spaces have no effect.	DELETE A B) DELETE A B)
/	/ (right slash key). The character following the slash is interpreted as a switch.	LIST/A)
;	; (command delimiter). Two or more lines can be on a line separated by semicolons. No commands are executed until a carriage return is entered.	CREATE A;LIST)
↑	↑ (shift and N key). The next carriage return is ignored as a line terminator. The ↑ must immediately preceed the carriage return).	RENAME A ALPHA,↑) B BETA
* -	To represent one character in a file name or extension. To represent variable numbers of characters in a file name or extension.	LIST T**) LIST--)
.)	Request or complement the extended prompt message format.	.) 14:34:54
@	Change of CLI command stream. Paired @ signs around a file name are understood to represent the contents of the file rather than the file name itself.	ASM@FOO@)
(,,,)	Repetition of commands or arguments. (Only commas can be used as argument separators when used with parentheses or brackets.)	(ASM, RLDR) D)
[,,]	User overlay definition.	See RLDR command in Chapter 3.
< >	In line expansion	MAC/V PART: ↑) <A,B,C,D>.SR)

A file name is made up of alphanumeric characters ($character is also acceptable), consisting of as many characters as desired (only the first ten are significant to the system).

A file must be opened before it can be accessed. Disk files can be opened fully for reading/writing or for reading only. Only one user can open a file fully but many can have access simultaneously, for reading only.

File Extensions

An extension can be appended to a file name and are used to make file names unique. Extensions are made up of alphanumeric characters ($ character is also acceptable) but only the first two are significant to the system. A period (.) separates the file name and the extension. For example:

File Name	Extension
PROG	.01
PROG	.02

In some cases the CLI itself appends extensions to a file name, indicating the type of data it contains. For example:

A.RB	-	Relocatable Binary file
A.SV	-	core image (SaVe file)
A.LS	-	LiSting file.
A.OL	-	OverLay file.

A user can also append extensions to a file, but care must be taken to use any of the system extensions properly.

Reserved Device Names

The following list indicates the names of devices used in CLI commands when addressing a device.

$CDR	Punched card reader; mark sense card reader
CTn	Data General cassette unit n, first controller
DK0	Data General fixed head NOVADISC, first controller
DPn	moving head disk unit n
$DPI	Input dual processor link

Reserved Device Names (Continued)

$DPO	Output dual processor link
$LPT	80 or 132-column line printer
MCAR	multiprocessor communications adapter receiver
MCAT	multiprocessor communications adapter transmitter
MTn	7 or 9-track magnetic tape transport n, first controller
$PLT	incremental plotter
$PTP	paper tape punch
$PTR	paper tape reader
QTY	asynchronous data communications multiplexor
$TTI	teletype or display terminal keyboard
$TTO	teletype printer or display terminal screen
$TTP	teletype punch
$TTR	teletype reader

The following list gives the names of second devices, i.e., second line printer (if applicable) etc.

$CDR1	second punched card reader
CT1n	Data General cassette unit n, second controller
DK1	Data General fixed head NOVADISC, second controller
$LPT1	second line printer
MCAR1	second multiprocessor communications adapter receiver
MCAT1	second multiprocessor communications adapter transmitter
MT1n	7 or 9-track magnetic tape transport n, second controller
$PLT1	second incremental plotter
$PTP1	second paper tape punch
$PTR1	second paper tape reader
$TTI1	second teletype or display terminal keyboard (only first keyboard is treated as a system console)
$TTO1	second printer or display terminal screen
$TTP1	second teletype punch
$TTR1	second teletype reader

File Attributes and Characteristics

File attributes are distinctive features of files which can be set and changed by the user via system calls and, in most cases, via the CLI.

File Attributes and Characteristics (Continued)

These attributes are:

P	permanent file, which cannot be deleted or renamed.
S	save file (core image).
W	write-protected file, which cannot be written.
R	read-protected file, which cannot be read.
A	attribute-protected file. The attributes of such a file cannot be changed. After the A attribute has been set it cannot be removed by a CLI command.
N	no resolution allowed. This attribute prevents a file from being accessed through a link.
?	first user-definable attribute (bit 9).
&	second user-definable attribute (bit 10).

File characteristics are distinctive features of files which cannot be changed by the user. The list of file characteristics is:

D	random file organization.
C	contiguous file organization.
I	accessible by direct I/O only. (Only SYS.DR and MAP.DR have this attribute).
L	link entry. Properly speaking, the L characteristic is given to directory entries rather than to the files themselves.
T	partition. This defines a file as being a partition.
Y	directory. This characteristic defines a file as being a directory.

Disk Partitions and Directories

RDOS permits the parceling of disk file space between several users on both a fixed and a semi-variable basis. Fixed parcels of contiguous disk file space are called secondary partitions. Secondary partitions are mutually exclusive subsets of total disk file space, the primary partition. File space within any partition can be allocated to users on a semi-variable basis. That is, users within a partition may be allocated mutually exclusive portions of that file space, and these portions may increase or decrease in size within the limits of total file space available in the parent partition. These variable parcels of file space are called subdirectories.

Each partition, primary and secondary, contains its own file directory. A file directory is a list called SYS.DR which records all file names and all subdirectory names within a given partition. The primary partition's SYS.DR also contains the names of all secondary partitions. Every subdirectory is a file directory listing

Disk Partitions and Directories (Continued)

the files in its file space. Like other directories, each subdirectory contains an entry called SYS.DR which is used for accessing its directory.

The primary partition can never be deleted. Subdirectories and secondary partitions, however, can be deleted. If a secondary partition is deleted, any subdirectories within that secondary partition are also deleted. Commands required to prepare partitions or subdirectories for system use and commands to delete them are given in Chapter 3.

COMMAND LINES

A command line can consist of one or more commands followed by a Carriage RETURN character (shown as) throughout this manual). A basic command line has one command.

Basic Command Line

Except for a number of simple commands that CLI executes directly, the command portion of the basic command line is interpreted by CLI by an associated file name i.e., the command ASM has the file name ASM.SV. For example if the user types the following.

```
ASM $PTR )
```

the CLI builds a file, called COM.CM (or called FCOM.CM in the foreground), containing the edited command line, and loads the save file that has the file name ASM.SV for execution.

Any additional file names besides the program name are used as arguments. In the example, $PTR is the file name of the paper tape reader from which a file is to be assembled. (In this example, the system would request the loading of $PTR twice since ASM.SV is a two-pass assembler.)

User action and CLI response are the same when a user wants to execute one of his own programs. For example, if a user has a save file named A.SV and he types

```
A )
```

The CLI builds a file containing the command line and calls the operating system to load the save file named A.SV for execution.

The user should be aware that it is possible to create a file containing standard CLI command lines used for some repetitive operation, i.e., a file containing CLI command lines necessary for the reporting, on some device, accumulated statis-

tical data. The INDIRECT@CONVENTION section in this chapter provides some valuable examples of this type of application.

Stacking Commands on a Command Line

A command line is executed by the CLI when the user presses the Carriage RETURN ()) key or the CTRL L keys on the teletypewriter.

A number of commands may be stacked on a given line for execution. They are separated by semicolons. For example:

```
CREATE A; LIST  A;  DISK;  DELETE  B )
```

The four commands are executed when the user presses). The CLI indicates execution of each command with the appropriate information, if any. At the completion of the entire command line, the CLI will prompt the user again with a ready message. For example, the previous command line might cause the response:

```
A.          0                    ←response to LIST A
LEFT:56,  USED:  200             ←response to DISK
R                                ←command line completed
```

Long Command Lines

Each command line is limited to being 132 characters in length, although this may be extended by typing the symbol ↑ immediately before pressing the). The up arrow causes the carriage return to be ignored. For example:

```
CREATE A B C; LIST; DISK; APPEND NEW.SR ↑ )
GAMMA.SR DELTA.SR )
```

is executed as if the following had been typed:

```
CREATE A B C; LIST; DISK; APPEND NEW.SR GAMMA.SR DELTA.SR )
```

In the previous example, the second line starts a new argument. Note that when a) is ignored, there is no delimiter between the last character on one line and the first character on the next line. Therefore, in the example the blank argument delimiter has been inserted before the up arrow.

The user can, of course, break an argument or command word into two lines:

```
CREATE A B C; LIST; DISK; APPEND NEW.SR, GAM ↑ )
MA.SR, DELTA.SR )
```

Long Command Lines (Continued)

is equivalent to:

```
CREATE A B C; LIST; DISK; APPEND NEW.SR, GAMMA.SR, DELTA.SR )
```

Ready Message Formats

The user can specify either a simple "R" ready message or an expanded message which also gives the time of day. The initial format is a simple "R", output at the completion of each CLI command. To complement the current type of ready message, the user types the command

```
.)
```

The following series of commands illustrates the complementing of the ready switch and the different formats in which the prompt may appear.

```
CREATE A
R               ← initial prompt is a simple "R"
.)              ← now the expanded version is requested
9:14:01         ← expanded ready response
R
LIST A          ← user issues a new command
A.   0          ← file A is listed
9:14:37         ← and expanded prompt is output
R
.)              ← ready switch is complemented
R
DELETE C )      ← user issues a new command
R               ← simple R response to this command
```

SWITCHES

Commands and their arguments may be modified by a series of switches pertaining to the command or argument. A switch is indicated by a right slash (/) followed by either a single letter or a single decimal digit.

Numeric Switches

Numeric switches specify the number of times the previous argument is to be repeated in the command line. For example:

```
RLDR $PTR/6 )
```

indicates that six relocatable binary tapes are to be loaded from the paper tape reader.

Numeric Switches (Continued)

Numeric switches are cumulative. The following commands are equivalent:

```
RLDR $PTR/1/0/3/2 )
RLDR $PTR/6 )
```

When used alone, the digit 1 in a numeric switch is the same as no switch. The following commands are equivalent:

```
RLDR $PTR )
RLDR $PTR/1 )
```

The digit 0 has no effect upon the number of times a file name is repeated if it appears in a list of numeric options. For example, the following commands are equivalent:

```
RLDR $PTR/6 )
RLDR $PTR/1/0/2/3 )
RLDR $PTR/1/2/3/0/0 )
```

However, when used alone, the 0 switch has the same effect as 1. For example, the following are equivalent commands:

```
RLDR $PTR/1 )
RLDR $PTR/0 )
RLDR $PTR )
```

The user should note the effect of applying a numeric switch to a CLI command. For example, the following are equivalent:

```
LINK ASM.SV/2 )
LINK ASM.SV  ASM.SV )
```

Letter Switches

Letter switches have distinct meanings that depend upon the command or argument with which they are associated. The detailed descriptions of each CLI command indicate the meanings of each letter switch that can be used in the command.

A letter switch that follows a command word is a global switch and applies to all arguments of the command line. A switch that follows an argument is a local switch and applies only to the particular argument. For example, the assembly command ASM has both a local and global switch, L (listing file). The command:

```
ASM  A B )
```

Letter Switches (Continued)

causes files A and B to be assembled but, by default, no listing is produced. The command:

ASM/L A B)

causes files A and B to be assembled, and a listing file named A. LS to be produced. The command:

ASM A B $LPT/L)

causes files A and B to be assembled and a listing of the assembly to be output to the line printer.

Whenever any listing file is a disk file, new listing file output is appended to the file (the previous listing file is not deleted).

ASTERISK (*) AND DASH (-) CONVENTIONS

Using the Asterisk

When referencing a file name, an asterisk can be used to represent any given character in the file name. For example, the command:

DELETE A**M)

will cause all four-character non-permanent files (in the current directory) without extensions that begin with A and end with M to be deleted. For example, files that have names like the following would be deleted: ATOM, ADAM, A22M, and A$RM.

The command LIST B*) causes a list of all two-character file names beginning with the letter B and having no extension to be listed.

In every case, use of the asterisk in referencing file names or extensions implies a fixed format. That is, use of only one asterisk implies that only one character in the file name may vary; use of two asterisks in a file name, as

LIST A**D)

causes certain four-character file names without extensions to be printed. Thus file names like ABCD, ACBD, A$3D would be listed while file names like ABC and ABCDE would not be listed. Use of a single asterisk, as

LIST *)

causes all single-character file names (without extensions) to be printed.

Using the Dash

Use of the dash in referencing file names implies a free format. That is, the dash may be used to represent any number of valid characters in a file name or extension (including no characters at all). For example, the command

```
LIST A-D )
```

causes all file names (without extensions) which begin with A and end with D in the current directory to be printed. Thus file names like ABCD would be listed as would names like AD, ABD, AND, ABC$D, etc.

Correspondingly, LIST -AB) would cause all file names ending in AB to be listed.

To list every non-permanent file in the current default directory, the command

```
LIST - . - )
```

would be issued. All file names, with and without extensions, would be listed. Similarly, it is possible to delete all files that are not protected with the single command DELETE - . -).

A device specifier cannot be used with either - or * conventions. An attempt to give a command such as:

```
LIST DP1: * . * )
```

causes the default directory to be searched for the name DP1: * . * which is never found.

Chapter 3 contains descriptions of each CLI command and each description indicates whether or not the asterisk convention can be used in file names. The dash convention may be used only when the asterisk convention is listed as being permitted.

INDIRECT @ CONVENTION

Paired @ signs around a file name are understood to represent the contents of the file rather than the file name itself.

Suppose a user regularly concludes each teletypewriter session by deleting listing files, checking the list of non-permanent files, and determining how much space he has left on disk. The command line for this would be:

```
DELETE -.LS; LIST; DISK )
```

INDIRECT @ CONVENTION (Continued)

These commands could be written into a file called END in the following way:

```
XFER /A $TTI END )          Transfer commands in ASCII
DELETE -.LS; LIST; DISK )   from TTI to file END. (User
("CTRL Z")                  terminates input with CTRL Z;
R                           CLI types our R.)
```

Then the command:

```
@END@ )
```

is equivalent to typing three commands.

As another example, suppose the user has five source programs called PART1, PART2, PART3, PART4, and PART5. He can then use the XFER command as shown above to build a file called TEST, containing the ASCII line:

```
PART1 PART2 PART3 PART4 PART5
```

If he issues the command:

```
ASM @TEST@ )
```

the five files are assembled.

The contents of a file on disk may, in turn, point to another file. As a simple example, suppose:

```
file A contains L@B@
file B contains I@C@
file C contains ST
```

Then the command:

```
@ A @ )
```

is equivalent to the command:

```
LIST )
```

An indirect file may itself contain nested indirect files. Suppose the contents of the three files were:

```
file A contains L
file B contains I
file C contains ST
```

INDIRECT @ CONVENTION (Continued)

Then the command

```
@A@@B@@C@ )
```

is equivalent to

```
LIST )
```

COMMA/PARENTHESIS/ANGLE BRACKET CONVENTION

CLI commands can be repeated with each of several arguments or argument strings by first separating the arguments or strings with commas and then by enclosing all the arguments within parentheses. Moreover, a series of commands can be made to use a common argument by placing the commands within parentheses. The general form of the convention is:

```
S1  (S2, S3, . . . , Sn)  Sn + 1 . . . )
```

where: S1 ... Sn+1 are strings within the command line sequence.

Use of the above generalized form results in a series of commands which are executed serially by the CLI:

```
S1  S2  Sn + 1 . . . )
S1  S3  Sn + 1 . . . )
    .
    .
    .
S1  Sn  Sn + 1 . . . )
```

Multiple sets of parentheses are permitted per command line. Parenthesis ( ) and angle brackets cannot be nested. The user must ensure to have an even number of parenthesis per line. Use of this convention to assemble a series of disk files is illustrated in the single command:

```
ASM (FILEA, FILEB, FILEC FILED) $LPT/L )
```

The CLI expands this command into the following set of three CLI commands:

```
ASM FILEA $LPT/L )
ASM FILEB $LPT/L )
ASM FILEC FILED $LPT/L )
```

COMMA/PARENTHESIS/ANGLE BRACKET CONVENTION (Continued)

An example of a single argument having several commands associated with it is:

```
(ASM,RLDR) MYFILE )
```

which is equivalent to:

```
ASM MYFILE )
RLDR MYFILE )
```

An example of the use of multiple parenthesis is:

```
MAC/N PART:(TEST,B,C,D).SR OVLY.SR MTØ:(Ø,],2,3)/L
```

The CLI expands this command to the following set of CLI commands:

```
MAC/N PART:TEST.SR OVLY.SR MTØ:Ø/L
MAC/N PART:B.SR OVLY.SR MTØ:1/L
MAC/N PART:C.SR OVLY.SR MTØ:2/L
MAC/N PART:D.SR OVLY.SR MTØ:3/L
```

Use of the comma-parenthesis convention is also permitted to complete and expand an argument name. Thus the command

```
FORT      SUB(1,2,3).FR )
```

is expanded to:

```
FORT      SUB1.FR )
FORT      SUB2.FR )
FORT      SUB3.FR )
```

Angle brackets (< >) are used for in-line expansion of code, and can be nested to any desired depth. Angle brackets may not be overlaped with parenthesis since this combination would result in a meaningless evaluation. (<>) or <( )> are legal but <(>) is illegal. Angle brackets are evaluated before parenthesis. The following is an example of the use of the angle bracket.

```
MAC/N PART: < A,B,C,D >.SR
```

The preceding example is interpreted by the CLI as follows:

```
MAC/N PART:A.SR PART:B.SR PART:C.SR PART:D.SR
```

Care must be taken not to overwrite disk files created by use of CLI commands, as in the following case:

```
FORT($CDR, $CDR,$CDR) $LPT/L
```

In this example, three separate FORTRAN compilations would be printed on the line printer, but only the third FORTRAN program input via the card reader would exist as an RB file on disk, $CDR.RB. The previous two disk files with this same name would have been overwritten by the third binary file. Users wishing to perform a series of compilations, assemblies, etc. are referred to the BATCH User's Manual.

FILE NAME SEARCHES

Any file directory may contain a number of entries having the same file name but different extensions, for example:

A.SR
A.RB
A.SV
A.32
A.XX
A.LS

File names used as arguments to most commands must specify both the appropriate file name and extension. Certain commands, however, will search for a file name with an appropriate extension appended to it, and if unable to find this name with the extension will search for the file without the extension. For example:

ASM A)

causes a search for the file named A.SR If A.SR is found, it is used as the source file for the assembly. Otherwise, a search is made for A. If instead the assembler source file A had an extension (.,.XX, etc.) a search is made for the file with that same extension and no other.

When A is found, the commands:

ASM A) ASM A.) ASM A.XX)

will each cause the assembler to produce a relocatable binary output file named A.RB. The CLI creates this name by adding the extension.RB (for relocatable binary) to the name of the source file.

Note: ASM A. will cause the search of A and not the default.

If the user types:

RLDR A)

A search is first made for A.RB and if not found, for A. The CLI creates an output file for the relocatable loader called A.SV, the extension used to signify a save file.

The command SAVE and MKSAVE also have a save file as output. In both cases, the CLI adds the extension SV to the name of the output file. If the user attempts to substitute his own extension, it will be ignored. For example:

```
SAVE A.XX )
```

causes a core image to be stored as a save file on disk. The name of the file will be A.SV. The extension XX is ignored.

The command MKABS has as input a save file. For example, if the user types:

```
MKABS A $PTP )
```

a search is made for A.SV and, if not found, for A.

To execute the file A.SV, the user types

```
A )
```

The CLI calls the operating system to load into memory the file called A.SV and to transfer control to its starting address. In this special case of loading a save file, the only search made is for the file name with the SV extension.

Most other commands require appropriate file name extensions to be given explicitly. If the user types:

```
DELETE A )
```

only the file named A will be deleted. Files such as:

```
A.SR
A.SV
A.RB
A.33
```

would not be deleted.

If the user types the command:

```
RLDR A FOO/S )
```

the /S switch indicates that the user wants the save file output of the loader to be named FOO.SV.

If a user gives his own extensions to file names, such as A.33, such files must be referenced with their file name and extension.

MESSAGES CONCERNING I/O

Some commands require manual operation of an I/O device. If the user issues such a command, he will receive a message prompting him on the proper action. For example, if the user issues the command:

```
XFER/A $PTR A.SR )
```

which requests that a source file be transferred from the paper tape reader to a disk file named A.SR, the system replies:

```
LOAD $PTR, STRIKE ANY KEY.
```

The user can then load the paper tape reader and strike any key on the console. The struck key is not echoed on the console.

When a series of files are to be transferred, assembled, or loaded from a device requiring manual intervention for each file, the message will be issued the appropriate number of times. For example, if the user issues the command:

```
APPEND NEWFILE $PTR/2 )
```

which requests that a file called NEWFILE be created from two files input from the paper tape reader, the following responses will occur:

```
LOAD $PTR, STRIKE ANY KEY.
LOAD $PTR, STRIKE ANY KEY.
```

The second message is typed out after the first file has been transferred.

SYSTEM CONSOLE BREAKS

There are three possible program breaks that can be generated at a system teletype or video display console: CTRL A, CTRL C, and CTRL F breaks. These breaks can cause a variety of actions to occur: terminate the current CLI command or BATCH job, terminate a program and save a snapshot of its core image, flush a BATCH job stream, and terminate a foreground program. For information concerning keyboard breaks in the BATCH monitor, consult the BATCH User's Manual, 093-000087. CLI and BATCH keyboard interrupts can be generated on either the foreground or the background console keyboards. CTRL F breaks are operative only on $TTI, the background console.

Pressing CTRL and A on a console keyboard causes an external interruption of user program activity. Control goes to a special break processing routine in either the current program or the closest higher level program with a break processing routine. The CLI residing at 0 has such a break processing routine. Thus, if no

lower level programs have such a routine, control will pass to the CLI upon a CTRL A break. Depressing CTRL A when the CLI is the current program will cause the CLI to cease its current operation and be ready for a new command. Note that the CTRL A interrupt will not halt output operations to such devices as the line printer if data is being spooled to these devices. Spooling operations must be halted by means of an SPKIL command, and spooling can be disabled via the console only by means of the SPDIS command.

When the CTRL A interrupt is issued in a multitask environment, any tasks which have outstanding system calls will have their system actions aborted when an interruption of the program takes place. The concept of program levels is discussed at length in Chapter 4, Program Swap and User Overlay Commands; of the RDOS User's Manual (093-000075) multitasking is discussed in Chapter 5 of the same manual.

Pressing CTRL and C on the keyboard, causes an eventual interrupt of a program and a file to be created and written as an image of core at the time of the interrupt. The word BREAK is typed by the Command Line Interpreter upon completion of this interrupt, and the name of the save file created will be BREAK.SV (for the background or FBREAK.SV (for the foreground).

Considerations similar to those given for CTRL A interrupts apply also to CTRL C breaks. The currently executing task becomes readied and loses program control as with a CTRL A break. Also as with CTRL A breaks, control passed to the next higher level program with a CTRL C interrupt processing routine. However, any tasks which have outstanding system calls will have those calls completed before the save file is created so that, upon resumption of the save file's execution, a rescheduling will occur which allocates CPU control the highest priority ready task.

Additional considerations also must be made, e.g., that all open channels will be closed and must be opened by the user before execution of the save file is resumed.

Under no circumstances are the CTRL A, C, or F break characters ever transmitted as input characters to a user program.

In a foreground/background environment, pressing CTRL F causes the foreground program to cease operation, returning system resources fully to the background program. For further discussion of the foreground/background environment, see Chapter 6 of the RDOS User's Manual (093-000075).

ERROR MESSAGES

When the user issues a command containing an error, an appropriate error message is typed.

When a user issues a command that is legal for some arguments and illegal for others, an error message is issued for each of the illegal arguments. The correct portions of the command are executed. For example,

```
R
CREATE B C D )                          ← create three empty files
R
XFER  $PTR A )                          ← transfer file from PTR to A
LOAD  $PTR, STRIKE ANY KEY.
R
CREATE A E )
FILE ALREADY EXISTS: A                  ← illegal argument A; legal
R                                                 argument E
LIST E )
E.                  0
R                                       ← E was created
```

When the CLI cannot respond to a user command, an error message does not necessarily result. For example, if the user requests list information on a non-existent file, the CLI responds to the LIST command with a ready message (R) only.

In general, error messages are quite explicit, giving the user sufficient information to correct his error easily.

A few samples are shown:

```
R
CREATE  A #A *A )
ILLEGAL FILE NAME:  #A
ILLEGAL FILE NAME:  *A
R

XFER  FOO  $PTR  )
FILE WRITE PROTECTED: $PTR
R
```

ERROR MESSAGES (Continued)

```
CREATE TEST;CHATR TEST W )
R
XFER SYS.DR TEST )
DIRECT I/O ACCESS ONLY:  SYS.DR
R

CHATR $LPT 0
ATTRIBUTE PROTECTED:  $LPT
R

MKSAVE $PTR TST.SV
LOAD $PTR, STRIKE ANY KEY.
FILE SPACE EXHAUSTED
R

XFER NONFILE NEWFILE
FILE DOES NOT EXIST:  NONFILE
R
```

A complete list of CLI errors is given in Appendix A.

CHAPTER 3

CLI COMMANDS

INTRODUCTION

This chapter defines and describes each of the CLI Commands which are organized in the following function order:

- Commands that Invoke Utilities
- File maintenance commands
- Foreground/Background commands
- System Generation commands
- Tuning file commands
- Spooling commands
- Time, Date, System name, Available disk block commands
- Bootstrap command
- Chain command
- Print current device directory command
- Communication command
- Higher level program command

The following conventions are used to define individual CLI command formats:

All upper case letters represent valid command line elements.

Items in a command line printed in lower-case indicate either command information or file names which must be supplied in the command line.

Elements enclosed in modified brackets, ⌈ ⌉, are optional. Stacked items enclosed in braces, $\left\{ \begin{matrix} \\ \\ \end{matrix} \right\}$, indicate a required selection from one of several alternate choices.

The ellipsis (...) is used to indicate that preceding file types or bracketed material may be repeated as desired.

The comma (,), plain brackets ([ ]), and right slash (/), up arrow ( ), asterisk (*), dash (-), parenthesis (()) are not conventions but are significant and necessary parts of any command line definition in which they are found.

Whenever a CLI command takes a filename argument, that argument can include directory specifiers except in commands MCABOOT and SAVE; these commands require arguments to be in the current directory only.

The following commands permit the use of the dash and asterisk conventions only when the filename argument is in the current directory: CLEAR, DELETE, DUMP, LIST, LOAD, UNLINK, MOVE, BUILD.

Note that before any system utilities (compilers, editors, etc.) can be called via the CLI, the dump tapes (088 prefix) containing these utilities must first be loaded onto disk via the CLI LOAD command.

COMMANDS THAT INVOKE UTILITIES

This section includes the CLI commands that are necessary for program development. The following list indicates the commands as they are presented.

ALGOL	-	Compile an ALGOL source file.
ASM	-	Assemble a source file.
BASIC	-	Execute a BASIC program.
BATCH	-	Execute a BATCH job stream.
CLG	-	Compile, load and execute a FORTRAN IV program.
DEB	-	Read a program and go to the debugger.
EDIT	-	Edit an ASCII file.
FORT	-	Compile a FORTRAN IV program.
FORTRAN	-	Compile a FORTRAN 5 program.
LFE	-	Update Library File.
MAC	-	Perform a macro assembly.
MEDIT	-	Multi-Editor
OEDIT	-	Edit a disk file.
OVLDR	-	Create as overlay replacement file.
RDOSSORT	-	RDOS Sort Merge
REPLACE	-	Replace overlay in replacement file.
RLDR	-	Perform a relocatable load.
SPEED	-	Edit an ASCII file.

Name: ALGOL (Compile an ALGOL source file.)

Format: ALGOL inputfilename {outputfilename...}

Purpose: To compile an ALGOL source file. Output may be a relocatable binary file, an intermediate source file, a listing file, or combination of all three. The command name, ALGOL, must be used in compiling ALGOL source programs; the name, ALGOL, cannot be changed.

Switches: By default, execution of the command produces an intermediate source file, inputfilename.SR (compiler output), and a relocatable binary file inputfilename.RB (assembler output). However, once assembly has been successfully completed, the intermediate source file is deleted. No listing is produced by the default command.

Global:

/A	-	Assembly is suppressed.
/B	-	Brief listing (compiler source program input only).
/E	-	Error messages from compiler are suppressed at the $TTO. (Assembler error messages are not suppressed.)
/L	-	Listing produced to inputfilename.LS.
/N	-	No relocatable binary file is produced.
/S	-	Save the intermediate source output file.
/U	-	Append user symbols to binary output file.

Local:

/B	-	Relocatable binary output directed to a given file name (overrides global /N).
/E	-	Error listing to given file name.
/L	-	Listing output directed to given file name (overrides global /L).
/S	-	Intermediate source output directed to given file name.

Asterisk: Not permitted.

Extensions: On input search for inputfilename.AL. If not found, search for inputfilename. On output, produce inputfilename.RB by default and other files with .LS or .SR extensions as determined by global switches.

Examples: ALGOL MAIN)

Produce relocatable binary file, MAIN.RB. No listing is produced.

Name: ALGOL (Continued)

Examples: (Continued)

```
ALGOL/E/B SUBR $ LPT/L )
```

Produce relocatable binary file SUBR.RB with a brief ALGOL source listing to the line printer. Suppress compiler error messages.

```
ALGOL/A RAY $PTP/S )
```

Do not invoke an assembly phase. Punch intermediate source output on high speed punch.

Name: ASM (Assemble a source file.)

Format: ASM filename$_1$ . . .[filename$_n$]

Purpose: To assemble one or more source files using the extended relocatable assembler. Output may be a relocatable binary file, a listing file, or both. The command name, ASM, must be used in assembling programs; the name, ASM, cannot be changed.

Switches: By default, output of an assembly is a relocatable binary file (no listing file).

Global:

/L	-	listing file is produced.
/N	-	no relocatable binary file is produced.
/U	-	user symbols are appended to the relocatable binary output.
/E	-	error messages are suppressed.
/S	-	skip pass 2. A BREAK is signaled after pass 1 permitting the user to save a version of the assembler that contains his own semi-permanent symbols.
/T	-	symbol table list is not produced as part of the listing. (Used when a listing is requested, which produces a symbol table by default.)
/X	-	produces cross referencing of symbol table. Symbol table output will contain page number - line number pairs for the symbol definition as well as every reference to the symbol within the assembly.

Local:

/B	-	relocatable binary output directed to a given file name. (overrides global /N)
/E	-	error messages are directed to a given file name.
/L	-	listing output directed to the given file name. (overrides global /L)
/S	-	skip this file on pass 2 of assembly. (This switch should be used only if the file does not assemble any storage words.)
/N	-	no listing of this file. (Used, when a listing is requested, to list a selected number of files to be assembled.)

Asterisk: Not permitted.

Name: ASM (Continued)

Extensions: On input, search for filename.SR. If not found, and the filename did not have an extension, search for filename.

On output, produce filename.RB for relocatable binary and filename.LS for listing (global L switch), where filename will be the name portion of the first source file specified without a /S, /L, or /B local switch given.

Examples: ASM Z)

causes assembly of source file Z, producing a relocatable binary file called Z.RB.

ASM/N/L A)

causes assembly of file A, producing as output a listing file called A.LS.

ASM (A,B,C,D,DK0:E, $PTR) $PTP/B $LPT/L)

causes separate assemblies of files A, B, C, and D from the default directory, of file E on fixed head disk unit 0, and of a paper tape mounted on the high speed reader. (The source program mounted on the reader would need to be reloaded since the assembler requires two passes.) Binary relocatable files for each source file are output on the high speed punch. Separate assembly listings are produced on the line printer.

CAUTION: issuing the following command would cause the loss of the first relocatable binary disk image:

ASM ($PTR,$PTR) $LPT/L)

Even though two distinct source files are read by the high speed reader, each relocatable binary produced is labeled $PTR.RB. Thus the first relocatable binary would be overwritten by the second one.

Name: ASM (Continued)

Examples: (Continued)

```
ASM /S/N ICODES )        <- no output. Automatic BREAK after
BREAK                       pass 1.
R
SAVE ASM.SV )            <- User can save the assembler with
                            the user's permanent symbols.

ASM /X EXAMP $LPT/L)     <- assemble EXAMP, giving cross
                            referenced symbol table, output
                            on the line printer.

                         <- Program Listing
```

```
  0001  EXAMP
01                          .TITLE EXAMPLE
02                          .EXTN SBROU
03                          .NREL
04                  ;  SAMPLE CROSS-REFERENCE PROGRAM
05                  ;
06  00000/020431  START:    LDA     0,C3
07  00001/040426            STA     0,CNT
08  00002/020430            LDA     0,MAGIC
09  00003/040425            STA     0,BITS
10  00004/125414            INC#    1,1,SZR
11  00005/000404            JMP     LOOP
12  00006/010421            ISZ     CNT
13  00007/010421            ISZ     BITS
14  00010/000413            JMP     OUT
15
16  00011/006422  LOOP:     JSR     @ SUBR
17  00012/101014            MOV#    0,0,SZR
18  00013/000410            JMP     OUT
19  00014/021000            LDA     0,A0,2
20  00015/000406            JMP     OUT
21  00016/021000            LDA     0,A0,2
22  00017/025377            LDA     1,A1,2
23  00020/006413            JSR     @ SUBR
24  00021/051376            STA     2,A2,2
25  00022/041000            STA     0,A0,2
26  00023/006410  OUT:      JSR     @ SUBR
27  00024/006002            .SYSTM
```

Name: ASM (Continued)

Examples: (Continued)

```
28  00025/064400            .RTN
29  00026/000400            JMP
30
31          000001  CNT:    .BLK    1
32          000001  BITS:   .BLK    1
33
34  00031/000003    C3      3
35  00032/000004    MAGIC:  1B13
36
37  00033/177777    .SUBR:  SBROU
38          000000  AO=     0
39          177777  A1=     -1
40          177776  A2=     -2
41
42                          .END
```

Cross-reference table.

```
  0002 EXAMP
AO      000000    1/19  1/21  1/25  1/38
A1      177777    1/22  1/39
A2      177776    1/24  1/40
BITS    000030/   1/09  1/13  1/32
C3      000031/   1/06  1/34
CNT     000027/   1/07  1/12  1/31
LOOP    000011/   1/11  1/16
MAGIC   000032/   1/08  1/35
OUT     000023/   1/14  1/18  1/20  1/26
SBROU   000033/X  1/37
START   000000/   1/06
.SUBR   000033/   1/16  1/23  1/26  1/37
```

Cross-referencing is accomplished by outputting symbol table and symbol referencing information to temporary disk files during assembly. A separate save file, XREF.SV, is called by the assembler to output the cross reference list. Note that all pages and lines of the assembler's listing are numbered for this purpose.

Name: BASIC (Invoke the BASIC interpreter.)

Format: BASIC

Purpose: To invoke the BASIC interpreter to execute a BASIC program. A BASIC save file, configured via BSG, must exist before this command can be executed. BASIC configuration and load procedures under RDOS are described in Appendix C of the BASIC User's Manual, 093-000065.

Switches: None.

Asterisk: Not permitted.

Name: BATCH (Execute a BATCH job stream.)

Format: BATCH { jobfile$_1$ . . . jobfile$_n$ } { outputfile/O } { logfile/G }

Purpose: To initiate the execution of one or more job files, i.e., to create a BATCH job stream. Each jobfile is an input file containing one or more user jobs, and each job is complete with job control commands and optional data sets. The line printer is the default output file, the card reader is the default input file, and the teletype printer is the default log file. By default, BATCH searches for job files with the .JB extension. Consult the BATCH User's Manual, 093-000087, for further information.

Switches:

Global: None.

Local: /O - define the output file, SYSOUT.
/G - define the log file.

Asterisk: Not permitted.

Example: BATCH DAILY.JB MT0:1 LOG/G)

This command defines a job stream with jobs serially input via first disk file DAILY.JB and then file 1 of MT0. Job log information is sent to a disk file named LOG, and the line printer becomes SYSOUT by default.

Name: CLG (Compile, load and execute a FORTRAN IV program.)

Format: CLG filename$_1$ { filename$_2$ . . . } { [filename . . .] . . . }

Purpose: To perform a FORTRAN IV compilation, loading, and execution of one or more FORTRAN IV source files. Output includes one or more intermediate source files, one or more relocatable binary files, and an executable save file. The save file is created by the relocatable loader, using the relocatable binary files and four of the FORTRAN IV libraries which must have been merged into a single library named FORT.LB (see the LFE merge command).

CLG differs from the FORT command which can produce a relocatable binary file, but cannot produce a save file and execute it. In addition, CLG can treat source input files individually, where some require loading, others assembly and loading, and still others compilation as well.

Switches: If a listing device is specified by a local switch but no global listing switches are given, listing of each FORTRAN IV compilation, each assembly, and the load map are output to the specified listing file.

Global:
- /B - brief listing (compiler source program input only).
- /M - loader map is suppressed. All compiler and assembler source programs are listed.
- /E - error messages from the compiler are suppressed. Assembler messages are not suppressed.
- /T - multitask CLG. (The multitask FORTRAN library, FMT.LB, must be available on disk.)

Local:
- /A - assemble and load this file only; do not compile.
- /L - listing output directed to the given file name.
- /O - load this file only; do not compile or assemble.

also, all other local switches found in ASM and RLDR.

Asterisk: Not permitted.

Extensions: On input, search for filename.FR; if not found, search for filename. If /A is specified, search for filename.SR. If not found, search for filename. If /N is specified, search for filename.RB; if not found, search for filename.

On output, produce temporary assembler source files, filename$_i$.SR (i = 1...n). Produce relocatable loader input files, filename$_i$.RB (i = 1...n). Produce save file filename$_1$.SV.

Name: CLG (Continued)

Examples: CLG/B MAIN $LPT/L)

Compile MAIN.FR (or MAIN), producing MAIN.SR, with the listing on the $LPT. Assemble MAIN.SR, producing MAIN.RB and delete MAIN.SR. Load MAIN.RB and FORT.LB, producing MAIN.SV. Execute MAIN.SV.

CLG/M/E PROG 1 PROG2 PROG3/A PROG4/O $LPT/L)

Compile PROG1.FR (or PROG 1), producing PROG1.SR. Assemble PROG1.SR, producing PROG1.RB and delete PROG1.SR. Compile PROG2.FR (or PROG2), producing PROG2.SR. Assemble PROG2.SR, producing PROG2.RB and delete PROG2.SR. Assemble PROG3.SR (or PROG3), producing PROG3.RB. Listings from each compilation and assembly are output on the line printer. Compilation error messages are suppressed. Load PROG1.RB, PROG2.RB, PROG3.RB, PROG4.RB (or PROG4), and FORT.LB, producing PROG1.SV with no loader map. Execute PROG1.SV.

CLG A B C)

Compile A.FR (or A), producing A.SR. Assemble A.SR, producing A.RB and delete A.SR. Compile B.FR (or B) producing B.SR. Assemble B.SR, producing B.RB and delete B.SR. Compile C.FR (or C), producing C.SR. Assemble C.SR, producing C.RB and delete C.SR. Load A.RB, B.RB, C.RB and FORT.LB producing A.SV. Execute A.SV.

Name: DEB (Load a program into memory and go to the debugger.)

Format: DEB filename

Purpose: To debug a program about to be executed. A symbolic debugger must have been loaded as part of the program save file, as described under the RLDR command. The DEB command transfers control to the debugger in this save file.

When debugging, memory can be examined, break points set, the program run, etc. After making any necessary changes in the program, the user can save the current core image of the program by issuing the $V Debug command and then save the core image under some file name, as described under the SAVE command.

Switches: None.

Asterisk: Not permitted.

Examples:

```
DEB A )                          ← Debug A.SV
1004/ADD 0 2 ADD 1 2)            ← Change program.
$V )
BREAK                            ← $V break issued.
SAVE A )                         ← Changed version (current core image)
                                   saved.
A )                              ← New attempt to execute.
```

Name: EDIT (Perform a string edit)

Format: EDIT { filename }

Purpose: To invoke a text editor to build a new source file or edit existing source files. This editor is described in DGC Document 093-000018. If the optional filename argument is given, the editor will automatically execute the command "UY filename" upon being loaded.

Switches: None.

Asterisk: Not permitted.

Example:

```
EDIT )
*                    ← Program is ready to accept commands.
.                    ← User issues editing commands.
.
.
*H$$ )               ← User terminates editing by pressing
                       the H key followed by two ESC keys.
R                      Return is made to the CLI.
```

Name: FORT (Compile a FORTRAN IV program.)

Format: FORT inputfilename { outputfilename1 ... }

Purpose: To compile a FORTRAN IV source file. Output may be a relocatable binary file, an intermediate source file, a listing file, or combinations of all three. The command name, FORT, must be used in compiling FORTRAN source programs; the name, FORT, cannot be changed.

Switches: By default, execution of the command produces an intermediate source file, inputfilename.SR (output of compilation) and a relocatable binary file, inputfilename.RB (output of assembly). However, once assembly has been successfully completed, the intermediate source file is deleted. No listing is produced by the default command.

Global:

- /A - assembly is suppressed (and intermediate source file deleted by default).
- /B - brief listing (compiler source program input only).
- /E - error messages from compiler are suppressed. (Assembler error messages are not suppressed.)
- /F - FORTRAN variable names and statement numbers are equivalenced to symbols acceptable to the assembler.
- /L - listing produced to inputfilename.LS.
- /N - no relocatable binary file is produced.
- /P - process only 72 columns per record or 72 characters per line (as in punched card images).
- /S - save the intermediate source output file.
- /U - user symbols are output during the assembly phase (must be used with /F).
- /X - compile statements with X in column 1.

Local:

- /B - relocatable binary output directed to given file name. (overrides global /N).
- /E - error messages are directed to a given file name. (overrides global /E).
- /L - listing output directed to given file name. (overrides global /L).
- /S - intermediate source output directed to given file name.

Asterisk: Not permitted.

Name: FORT (Continued)

Extensions: On input, search for filename.FR. If not found, search for filename. On output, produce filename.RB by default and other output files with .LS or .SR extensions as specified by global switches.

Examples: FORT/L MAIN)

produce relocatable binary file MAIN.RB with both a compiler and an assembler listing to file MAIN.LS.

Name: FORTRAN (Compile a FORTRAN 5 program.)

Format: FORTRAN inputfilename { outputfilename$_1$ ... }

Purpose: To perform a FORTRAN 5 compilation. Output may be relocatable binary file, a listing file, or both. The command name FORTRAN must be used in all FORTRAN 5 compilations, and cannot be changed.

Switches: By default, execution of the command produces a relocatable binary file, inputfilename.RB. No listing is produced by default.

Global:

/B - brief listing (compiler source program input only).
/C - check syntax only.
/D - debug aid. Line number and program name output on all run time error messages.
/L - listing produced.
/P - process only 72 columns per record or 72 characters per line (as in punched card images.)
/S - generate code to do subscript checking.
/X - compile statements with X in column 1.

Local:

/B - relocatable binary output directed to given file name.
/E - error output is directed to a given file name.
/L - listing output directed to given file name (overrides global /L).

Asterisk: Not permitted.

Errors: See FORTRAN 5 User's Manual, 093-000085.

Extensions: On input, search for filename.FR. If not found, search for filename. On output, produce inputfilename.RB by default and listingfile.LS if global switch /L is given.

Examples: FORTRAN/L MAIN)

produce relocatable binary file MAIN.RB with both a compiler and a listing on file MAIN.LS.

FORTRAN BMARK FLIST/L)

compile FORTRAN 5 source program BMARK and output a compiler listing and a source listing to disk file FLIST.

Name: LFE (Update library files.)

Format:

LFE A inputmaster {arg$_1$... } {listing-device/L}

LFE A/M inputmaster$_1$ {...inputmaster$_n$} {listing-device/L}

LFE D inputmaster {outputmaster/O} arg$_1$ {...arg$_n$}

LFE I inputmaster {outputmaster/O } file$_1$ [...file$_n$}

LFE M {outputmaster/O} inputmaster$_1$ {...inputmaster$_n$}

LFE N {outputmaster/O } file$_1$ {...file$_n$}

LFE R inputmaster {outputmaster/O} arg$_1$ file$_1$ {...arg$_n$ file$_n$}

LFE T inputmaster {listing device/L } {inputmaster$_2$...}

LFE X inputmaster$_1$ arg$_1$ {...arg$_n$}

Purpose: To edit and analyze library files, which are sets of relocatable binary files having special starting and ending blocks and which are usually designated by the extension .LB.

A, D, I, M, N, R, T and X are keys designating LFE functions. inputmaster and outputmaster represent library files; arg's represent logical records on the library files; file's are update files. The name LFE cannot be changed.

Action taken by the LFE depends upon the function given in the command:

A - analyze - Analyze global declarations of inputmaster of a series of inputmasters, or of logical records specified from one inputmaster. Output is a listing with symbols, symbol type, and flags; no new output library file is created. Default output device is $LPT.

D - delete - Delete logical records, specified by args from inputmaster, producing outputmaster. Default output file is D.L1.

I - insert - Insert relocatable binary files, merging with logical records of inputmaster in the manner described under Switches. Default output file is I.L1.

Name: LFE (Continued)

Purpose: (Continued)

M	-	merge	-	Merge library files (inputmasters) into a single library file named outputmaster. Default output file is M. L1.
N	-	new	-	Create new library file, outputmaster, from one or more relocatable binary files given by files. Output file is N. L1 by default.
R	-	replace	-	Replace logical records in inputmaster by relocatable binary files, producing outputmaster. Arguments are paired, with the first being the logical record and the second the relocatable binary file that replaces the logical record. Default output file is R. L1.
T	-	titles	-	Output to the listing device ($LPT by default) the titles of logical records on inputmaster.
X	-	extract	-	Extract from library file inputmaster one or more relocatable binary files given by args. Output is one or more relocatable binary files named args. RB.

Switches:

Key: /M - multiple input library files. The switch modifies the A function and causes all library file names following, except the listing file, to be analyzed as one library.

Local: /A - Insert after. The switch modifies a logical record in an I function command line. Arguments following the switch are inserted after the logical record whose names precedes the switch. When neither a /A or /B switch is given, inserts are made at the beginning of the new library file.

Name: LFE (Continued)

Switches: (Continued)

Local:

/B - Insert before. The switch modifies a logical record in an I function command line. Arguments following the switch are inserted before the logical record whose name precedes the switch. When neither a /A nor /B switch is given, inserts are made at the beginning of the new library file.

/E - error listing directed to given filename.

/F - form feed. This switch, used only with local /L, causes the analysis of each relocatable binary to be printed on a separate page.

/L - listing file. The switch modifies the name of a file to be used as listing output in the A or T function command line. (The $LPT is used by default.)

/O - output library file (overrides default settings in D, I, M, N, and R).

Asterisk: Not permitted.

Extensions: If no extensions for inputmaster or file are given in the command, LFE searches for inputmaster.LB or file.RB respectively. If not found, LFE searches for inputmaster or file.

Example: LFE N $PTP/O A.RB C.RB)

Create a library file, output to the punch from two disk files, A.RB and C.RB.

Name: MAC (Perform a macro assembly.)

Format: MAC filename$_1$ {...filename$_n$}

Purpose: To assemble one or more source files using the macro assembler. Output may be a relocatable binary file, a listing file, or both. The command name, MAC, must be used in performing macro assemblies of programs; the name MAC cannot be changed.

Switches:

Global: By default, output of an assembly is a relocatable binary file (no listing file).

- /A - add semi-permanent symbols to cross reference.
- /E - prevent error messages from appearing on the error file (unless there is no listing file). The console is the default error file.
- /F - a form feed is generated or suppressed as necessary to produce an even number of assembly pages. By default, a form feed is always generated.
- /L - listing file is produced. Listings include a cross reference of symbol table. MACXR.SV must be available on disk.
- /K - MAC.ST is kept at the end of assembly. By default, MAC.ST is deleted at that time.
- /N - no relocatable binary file is produced.
- /O - override all listing suppression controls.
- /S - skip pass 2 and save a version of the assembler's symbol table and macro definitions in file MAC.PS (deleting the old MAC.PS, if any).
- /U - user symbols are included in the relocatable binary output.
- /Z - the DGC proprietary license heading is printed at the top of assembly and cross-reference listing pages. By default, the proprietary license heading is not printed.

Local:

- /B - relocatable binary output is directed to the given file name (overrides global /N).
- /E - error output is directed to a given file name.
- /L - listing output is directed to the given file name (overrides global /L).
- /S - skip this file on pass 2 of assembly. (This switch should be used only if the file does not assemble any storage words. Macro definition files can be skipped on pass 2.)

Name: MAC (Continued)

Asterisk: Not permitted.

Extensions: On input, search for filename.SR. If not found, and the filename did not have an extension, search for filename.

On output, produce filename.RB for relocatable binary and filename.LS for listing (global L switch), where filename will be the name portion of the first source file specified without a /S, /L, or /B local switch given.

Examples: MAC Z)

causes assembly of source file Z, producing a relocatable binary file called Z.RB.

MAC LIB/S (A,B,C) $LPT/L)

produces relocatable binary files A, B, and C. File LIB contains macro definitions and thus is skipped during the second pass. A listing and cross-referenced symbol table are produced on the line printer.

Name: MEDIT (Multi Editor)

FORMAT: MEDIT terminals {ticks }

Purpose: To allow several users to perform editing at the same time. Each user is served as though he were the only user, except for a possible slight degradation in response time. The editor is described in DGC document 093-000018.

terminals is the number of text-editing terminals to be supported, in decimal. The programmer can only access a terminal within the specified range, e.g., if the programmer specifies 10 terminals, only 0-9 can be accessed.

ticks is an optional argument specifying the number of system clock ticks at which task rescheduling will be forced.

Switches: None

Asterisk: Not Permitted

Example:

```
MEDIT 6 )
*                    ← Program is ready to accept commands,
.                    ← User issues editing commands
.
.
```

Name: OEDIT (Edit a disk file.)

Format: OEDIT filename

Purpose: To invoke the octal editor in order to examine and modify in octal, decimal, or ASCII any location in any user file. See the Octal Editor manual, 093-000084.

Switches: None.

Asterisk: Not permitted.

Extensions: The octal editor searches for whatever file name and extension are given.

Example:

```
OEDIT FOO.SV )        ← If OEDIT finds FOO.SV, the editor
                        gives a carriage return/line feed.
14/ 016762            ← User proceeds with editing.
.
.
.
HOME                  ← To return to the CLI, user types H.
R                       OEDIT echoes OME and user is at
                        command level.
```

The current value of NMAX for any save file can be found by issuing the OEDIT command

404-16/ nnnnnn

The value nnnnnn is returned by OEDIT, and indicates the contents of USTNM of the User Status Table; USTNM contains the current value of NMAX (See Chapter 5 of the RDOS User's Manual, 093-000075). Note that all save files are core images minus the first 16 words of memory (system save files are an exception to this). Thus to reference a memory location n is a save file, n - 16 must be applied.

Name: OVLDR (Create an overlay replacement file.)

Format: OVLDR save file name overlay descriptor /N overlay list
{ overlay descriptor/N overlay list } . . .

Purpose: To create an overlay replacement file which, by means of the REPLACE command, will substitute overlays in the current overlay file with overlay replacements. The overlay replacement file created by OVLDR will bear the same name as the save file (or overlay file) with which it corresponds, but the replacement file will have the extension .OR. The original save file must have been loaded with a symbol table. The name OVLDR cannot be changed.

Switches:

Global:

/A - produce an additional symbol table listing with symbols ordered alphabetically. (The local /L switch must also be given.)

/E - error messages are output to the console when a listing file has been specified (local/L). By default when a listing file has been specified, error messages to the console are suppressed. (If no listing file is given, error messages are always output to the console.)

/H - cause all numeric output to be printed in hexadeciamal (radix 16). By default, output is in octal.

Local:

/E - error messages output to given file name.

/L - listing of symbol table is produced on the output file whose name precedes this switch. The table will list symbols in numeric order.

/N - overlay descriptor name precedes this switch immediately, and collection of overlay replacement relocatable binaries follows.

Asterisk: Not permitted.

Extension: A search is made for save file name with the .SV extension. If not found, a search is made for that file without the .SV extension. The output overlay replacement file will bear the name save file name.OR.

Name: OVLDR (Continued)

Example: OVLDR EXAMPLE.SV MODULE1/N NEWMODULE)

This example creates an overlay replacement file, EXAMPLE.OR. In this replacement file is an overlay with binary NEWMODULE which will replace the overlay with symbolic name MODULE1 in EXAMPLE.OL. The overlay replacement will not occur until the CLI command REPLACE is executed.

Name: RDOSSORT (RDOS Sort/Merge)

Format: RDOSSORT inputfilename$_1$ { inputfilename$_n$ } { outputfilename/O }
key$_1$ { key$_n$ } { arguments }

Purpose: Perform the following functions: rearrange the records in a disk or tape file, delete records from a disk or tape, reformat the records in a disk or tape file, produce sorted address files for a disk file, merge disk, or tape files into a single disk or tape file. A complete discussion of sort/merge is contained in the RDOS Sort/Merge User's Manual (093-000108).

Switches:

Global:

/D	-	Sort data in descending order (ascending is otherwise assumed).
/M	-	Merge operation is requested.
/N	-	No listing of statistics.

Local:

/B	-	Name of file containing lower field limit.
/D	-	Delete input file after sort completed. (overriden if no output file specified).
/K	-	Filename of sorted keys.
/L	-	Listing filename for sorting statistics (default is console output device).
/O	-	Filename of sorted data file (outputfile).
/S	-	User specified collating sequence filename.
/R	-	Record size specifier (in decimal).
/U	-	Name of file containing upper field limit.
/W	-	User specified work or temporary files (up to six may be specified).

Example: The user is directed to the sample jobs contained in the RDOS Sort/Merge User's Manual (093-000108).

Name: REPLACE (Replace overlays in overlay file.)

Format: REPLACE save file name

Purpose: To cause the actual replacement of overlays in an overlay file named save file name.OL with overlays in a previously created overlay replacement file, save file name.OR. The overlay replacement file was created by means of the CLI command OVLDR. Actual replacement will occur as soon as there are no outstanding overlay load requests.

Switches: None.

Asterisk: Not permitted.

Example: REPLACE ABC)

This command causes overlays in file ABC.OR to replace overlays in file ABC.OL. CLI command OVLDR must have been used to create overlay replacement file ABC.OR.

Name: RLDR (Perform a relocatable load.)

Format:

(1) RLDR root binary... [root binary... / [overlay binary..., ...]]...

(2) RLDR NREL partition/F ZREL partition/Z root binary... [root binary... / [overlay binary..., ...]]...

(3) RLDR/X root binary ... [overlay binary...]

Purpose: To create a save file from the loading of relocatable binary files, and optionally to create an overlay file from other relocatable binaries.

By default, all relocatable loads are loads in the background. Foreground loads for unmapped systems are indicated when memory partition addresses with local switches /F and /Z are found in the load command line (second format). The partition addresses define the starting ZREL and NREL addresses of the foreground load. The selected NREL partition address must be equal to $16_8 + n * 400_8$ where n is a positive integer. If the given NREL partition address is not of this form (i.e., n is not an integer), the loader will adjust the NREL partition address by rounding n upwards to the next integer value.

All relocatable loads using the global /X switch (third format) are system loads. In system loads there must be only one overlay area, and each overlay in the overlay file must be no more than 256 decimal words in length. No overlay can define ZREL addresses. Overlays are loaded as though NMAX were reset to zero; thus address constants within overlays are resolved relative to zero. No unresolved external references to page zero must remain when the last root binary in the command string has been loaded; no root binary in the command string may follow the bracketed overlay definition.

By default, the system library (SYS.LB) is searched after the last binary file name. This causes modules from SYS.LB to be placed at the end of the root program. By default, each relocatable load produces no symbol table listing and positions the symbol table so that the end of the symbol table coinsides with the first address not loaded by the program.

Name: RLDR (Continued)

Purpose: (Continued)

The left bracket, used to define the start of an overlay area, is equivalent to a SHIFT K on the console keyboard. Right bracket is equivalent to SHIFT M.

Switches:

Global:

/A - produce an additional symbol table listing with symbols ordered alphabetically. (The local switch /L must also be given.)

/C - causes loading to be compatible with RTOS/SOS conventions, i.e., INMAX is set at 440, save file starts at 0 (as with /Z global switch), USTSA contains the program starting address, and SYS.LB is not searched unless specified by the user.

/D - causes loading of symbolic debugger DEBUG III from SYS.LB. The symbolic debugger IDEB will be loaded instead of DEBUG III only if IDEB.RB appears some where in the RLDR command line. Note that the symbol table will not be written to the save file unless a /D switch is given.

/E - error messages are output to the console when a listing file has been specified (local/L). By default when a listing file is specified, error messages to the console are suppressed. (If no listing file is given, error messages are always output to the console.)

/H - causes all numeric output to be printed in hexadecimal (radix 16). By default, output is printed in octal.

/M - suppress load map and all console output. This speeds loading on systems with teletype consoles, but should be used with caution since it suppresses all load error messages.

/N - inhibits a search of the system library SYS.LB. By default, the search is performed.

/S - symbol table left at the high end of memory (must be used with /D).

/X - allow up to 128_{10} system overlays.

/Y - allow up to 256_{10} system overlays.

Note: /X and /Y are mutually exclusive switches.

Name: RLDR (Continued)

Switches: (Continued)

Global: (Continued)

/Z - start save file at location zero. Note that the /Z switch should be used with caution. A save file produced using the /Z switch cannot be executed properly under RDOS. Its primary purpose is to enable loading of routines that use page zero locations 0 - 15. The save file can then be output using MKABS/Z to produce an absolute binary that can be read in the stand-alone mode, using the binary loader or HIPBOOT (see Appendix E of RDOS User's Manual, 093-000075).

Local: By default, the first input file name is used with a .SV extension to form the name of the output file, and the first input file is used with the .OL extension to form the overlay file name. By default, user symbols are not loaded.

/C - preceding octal value specifies the number of channels required. This value is placed in USTCH of the User Status Table and overrides any channel number appearing in a .COMM TASK statement.

/E - error messages output to given file name.

/F - preceding octal value is the foreground NREL partition address.

/K - preceding octal value specifies the number of tasks required. This value overrides any task number appearing in a .COMM TASK statement.

/L - listing of symbol table is produced on the output file whose name precedes the switch. The table will list symbols in numeric order.

/N - NMAX, the starting address for loading a file, is forced to an absolute address given by the octal number preceding the switch. The specified value must be higher than the current value of NMAX when the argument is encounted.

/S - save file is given the preceding file name with the .SV extension. Overlay file is also given the preceding file name but with the .OL extension.

/U - user symbols are loaded from the relocatable binary file which precedes the switch.

Name: RLDR (Continued)

Switches: (Continued)

Local: (Continued)

/V - create a virtual overlay file for use in extended address space.

/Z - preceding octal value is the foreground ZREL partition address.

Asterisk: Not permitted.

CAUTION: The .COMM TASK statement or /C and /K local switches must be used in multitask programming to avoid fatal load error "LOAD OVERWRITE 17."

Extensions: A search is made for each input file with the name root binary.RB. If not found, then a search is made for root binary.

The default output file name will be root binary$_1$.SV. Otherwise, the output file name will be the file name preceding the switch /S with the .SV extension appended.

Examples: RLDR A B C DP2:D)

causes files A, B, and C from the default directory and D from the disk pack unit 2 to be loaded to produce save file A.SV on the default directory.

RLDR A/S $PTR)

causes the file in the paper tape reader to be loaded to produce a save file named A.SV.

RLDR /D A B C)

causes files A, B, and C to be loaded, together with the symbolic debugger to produce save file A.SV.

RLDR $LPT/L A 440/N B)

causes A and B to be loaded. Loading of B starts at 440_8. A numeric core map is printed on the line printer.

Name: RLDR (Continued)

Examples: (Continued)

RLDR R0 [A, B, C, D] R1 R2 [E, F G, H])

causes R0, R1, and R2 to be loaded as a root program with two overlay areas. Overlay area 0 is situated between R0 and R1; overlay area 1 follows binaries R1 and R2. Binaries A, B, C and D constitute overlay numbers 0, 1, 2, and 3 of node 0. Likewise, E becomes overlay 0 of node 1. F and G become overlay 1 of node 1, and H becomes overlay 2 of node 1. The overlay file contains both overlay segments and is named R0.OL. (See Chapter 4, User Overlays, in the RDOS User's Manual, 093-000075 for a further discussion of this example.)

RLDR X)

File X in the current directory is loaded to produce save file X.SV. All loader messages (except a load map) are output to the console; there is no load map.

RLDR X $LPT/L)

File X is loaded to produce X.SV. All loader messages (including the load map) go to the line printer.

RLDR/E X $LPT/L)

File X is loaded to produce X.SV. All loader messages go to $LPT. Additionally, all messages (except the load map) go to the console.

Name: SPEED (Superedit)

Format: SPEED { filename }

NSPEED { filename }

Purpose: To edit ASCII text. Included are the following capabilities: multi-buffer editing, multiple I/O files, macro programming, numeric variables.

The first format SPEED is applicable for the ECLIPSE line of computers and the second format NSPEED is applicable for the NOVA line of computers.

SPEED is described in the DGC document 093-000111.

Switches: None

Asterisk: Not permitted.

Example:

```
SPEED )
!                    ← Program is ready to accept commands.
.
.                    ← User issues editing commands.
.
!H$$ )               ← User terminates SUPEREDIT and returns
                       to CLI.
```

FILE MAINTENANCE COMMANDS

This section includes the CLI commands that are necessary for file maintenance. The following lists indicate the commands, in various categories, as they are presented.

File Creation Commands

CCONT	-	Create a contiguous file.
CRAND	-	Create a random file.
CREATE	-	Create a sequential file.

File Handling Commands

APPEND	-	Concatenate two or more files.
BPUNCH	-	Punch a binary file.
BUILD	-	Build a file containing file names.
CHATR	-	Change a files attributes.
CHLAT	-	Change link access attributes.
CLEAR	-	Set file use count to zero.
DELETE	-	Delete a file or series of files.
DUMP	-	Dump one or more files.
ENDLOG	-	Close the log file (stop recording console output.)
FILCOM	-	Compare two files.
FPRINT	-	Print a disk file in bytes, decimal, hexedecimal or octal.
LOAD	-	Reload dumped files.
LOG	-	Open the log file (record all CLI dialogue).
MKABS	-	Make an absolute file from a core image file.
MKSAVE	-	Make a save file from an absolute binary file.
MOVE	-	Move files from one directory to another.
PRINT	-	Print a file on the line printer.
PUNCH	-	Copy an ASCII file to the high speed paper tape punch.
RELEASE	-	Release a device from the system.
RENAME	-	Change the file name.
REV	-	To display the revision level of a save file.
SAVE	-	Save the core image as a save file.
TYPE	-	Output file contents on the system console.
UNLINK	-	Delete a link entry name.
XFER	-	Copy the contents of a file into another file.

Directory Commands

CDIR	-	Create a subdirectory.
CPART	-	Create a secondary partition.
DIR	-	Change the current default directory.
EQUIV	-	Assign a new name to a directory specifier.

Directory Commands (Continued)

GDIR	-	Print the current directory device name.
INIT	-	Initialize a directory or device.
LINK	-	Create a link to a file on the same or another directory.
LIST	-	List file directory information.
MDIR	-	Get the name of the current master directory.

Name: CCONT (Create a contiguous file.)

Format: CCONT filename$_1$ blockcount$_1$ { filename$_n$ blockcount$_n$ } ...

Purpose: To create one or more contiguous files. The entry specifies a file with a length given in decimal disk blocks by blockcount, and no attributes. Disk storage is allocated in increment of 256_{10} words each.

Switches: None.

Asterisk: Not permitted.

Examples: CCONT ALPHA 20)

Create a contiguous file, ALPHA, in the default directory, and allot disk storage for the file.

CCONT TEST 100 DP0:TEST1 51)

Create two filenames, TEST in the default directory and TEST1 in primary partition DP0.

Name: CRAND (Add an entry for a randomly organized file.)

Format: CRAND filename$_1$ { filename$_2$ ... }

Purpose: To create a randomly organized file. The entry specifies a file of zero length and no attributes.

Switches: None.

Asterisk: Not permitted.

Example: CRAND RANDFILE)

Create a randomly organized file named RANDFILE in the current directory. The file is of zero length and has no attributes. The size of the file will grow as required during use.

Name: CREATE (Create a sequential file.)

Format: CREATE filename$_1$ { filename$_2$ ... }

Purpose: To create a sequential file. The file created has a file of zero length and no attributes.

Switches: None.

Asterisk: Not permitted.

Examples: CREATE ALPHA)

Create a filename, ALPHA, in the default directory.

CREATE TEST TEST1 DP0:TEST2)

Create three file names, TEST, and TEST 1 in the default directory and TEST2 in the primary partition DP0.

Name: APPEND (Concatenate two or more files.)

Format: APPEND newfilename oldfilename$_1$ . . .[oldfilename$_n$]

Purpose: To create a new file, consisting of a concatenation of one or more old files in the order in which their names are listed as arguments. The old files are not changed by the command.

Switches: None.

Asterisk: Not permitted.

Examples: APPEND COM.SR COM1 COM2 COM3 COM4)

causes creation of the file COM.SR containing the contents of files COM1, COM2, COM3, and COM4 in that order.

APPEND DP1:ALL.LB A.LB B.LB DP0:C.LB)

causes creation of the file ALL.LB on disk pack unit 1 containing the contents of files A.LB and B.LB from the default directory and C.LB from disk pack unit 0.

Name: BPUNCH (Punch a binary file.)

Format: BPUNCH filename$_1$ { filename$_2$ . . . }

Purpose: To punch a given file or files in binary on the high speed punch. The command is the equivalent of a series of XFER commands:

XFER filename$_1$ $PTP; . . . ;XFER filename$_n$ $PTP)

The files may come from any device.

Switches: None.

Asterisk: Not permitted.

Examples: BPUNCH FEE.SR FI.SR FO.RB FUM.RB)

This command causes source files FEE and FI, and relocatable binary files FO and FUM to be punched on the high speed punch.

BPUNCH $PTR)

This command causes a paper tape duplicate of the tape in the high speed reader to be punched.

Name: BUILD (Build a file containing names that match specifiers.)

Format: BUILD outputfilename {filename$_1$...}

Purpose: To allow the inclusion of all file names from the current directory who match filename into the file named in outputfilename. If the extension is omitted the names of all files with no extensions will be incorporated into outputfilename.

Switches:

Global:

- /A - add permanent file
- /K - no links
- /N - do not include extensions associated with files in the output file.

Local:

- /A - build a file this day or after. Preceding argument is of the form MM-DD-YY where MM and DD can be one or two digits.
- /B - build a file before this day. Preceding argument is of the form MM-DD-YY where MM and DD can be one or two digits.
- /N - build a file to contain any files that do not match this name

Asterisk: Permitted

Example: BUILD ABC -.SR FORT -.-

causes the name of every file in the current directory with the .SR extension and files beginning with FORT and any extension to be written into file ABC.

Name: CHATR (Change a file's attributes.)

Format: CHATR filename$_1$ attributes$_1$ [...filename$_n$ attributes$_n$]

Purpose: To change, add, or delete resolution file attributes of a given file. All current resolution attributes of the file are replaced by those given in the attributes argument.

Switches: None.

Asterisk: Permitted in attributes argument only.

Attributes:

N	-	not a resolution entry (cannot be linked to).
P	-	permanent file. Cannot be deleted or renamed.
R	-	read-protected file. Cannot be accessed for reading.
S	-	save file. Having been set, this attribute cannot be removed.
W	-	write-protected file. Cannot be altered.
0	-	no attributes.
*	-	represents current file attributes (dash convention is not used).
?	-	user-defined attribute.
&	-	user-defined attribute.

When several attributes are specified for a given file name, they must be given as a single argument. Attributes may be listed in any order in the argument.

The following attributes and characteristics cannot be set or used as CLI arguments:

A	-	attribute-protected file.
C	-	contiguously organized file.
D	-	randomly organized file.
I	-	accessible by direct I/O only.
T	-	partition (the file is a partition).
Y	-	directory (the file is a directory).

Examples:

```
CHATR A 0 B R )
```

deletes all attributes of A and causes B to be read-protected.

```
CHATR A.SV W )
```

causes A.SV to be a write-protected save file. If A.SV had had other previous attributes, these would be deleted.

Name: CHLAT (Change link access attributes.)

Format: CHLAT filename$_1$ attributes$_1$ ... { filename$_n$ attributes$_n$ }

Purpose: To change, add, or delete file link access attributes. All current link access attributes of the file are replaced by those given in the attributes argument.

Switches: None.

Asterisk: Permitted in attributes argument only.

Attributes:

N	-	not a resolution entry (cannot be linked to).
P	-	permanent file. Cannot be deleted or renamed.
R	-	read-protected file. Cannot be accessed for reading.
S	-	save file. Having been set, this attribute cannot be removed.
W	-	write-protected file. Cannot be altered.
0	-	no attributes.
*	-	represents current file attributes (dash convention is not used).
?	-	user-defined attribute.
&	-	user-defined attribute.

When several attributes are specified for a given file name, they must be given as a single argument. Attributes may be listed in any order in the argument.

The following attributes and characteristics cannot be set or used as CLI arguments:

A	-	attribute-protected file.
C	-	contiguously organized file.
D	-	randomly organized file.
I	-	accessible by direct I/O only.
T	-	partition (the file is a partition).
Y	-	directory (the file is a directory).

Examples: See those given for CHATR.

Name: CLEAR (Set file use count to zero.)

Format: CLEAR { filename$_1$ ... }

Purpose: To clear the file use count in one or more SYS.DR entries. This command may only be issued in a background-only environment from level zero. This command is of use in the event of a system failure when several files were left open. Such files could not be renamed, deleted, etc., until their usage counts were set to zero. Another common use of CLEAR is prompted when a system is shut down without releasing the master device. This causes the "PARTITION OCCUPIED" message to be issued by HIPBOOT; CLEARing the partition's SYS.DR causes the message to be omitted.

Switches:

Global:

/A - all files (except CLI.OL, CLI.ER, SYS.OL, LOG.CM, SYS.TU) in the current directory have their usage counts set to zero.

/D - Device use counts are set to zero.

/V - verify with a list, on $TTO of files cleared.

Local: None.

Asterisk: Permitted only when filename argument is in the current directory.

Examples: Due to a hardware failure, an RDOS system needs to be reinitialized. One or more files were open at the time of the failure, and the usage counts of these files must be set to zero so that the files can be deleted or renamed. The command

```
CLEAR/A )
```

is issued.

```
CLEAR  DP0:ABC:D )
```

causes file D's use count to be set to zero.

Note: If the error message FILE IN USE: CLI.OL is received during an attempt to dump CLI.OL, this file must be cleared explicitly, "CLEAR CLI.OL) ", only when this particular CLI.OL is not in use (e.g., it is in another partition or you are dumping under BATCH control.).

Name: DELETE (Delete a file or series of files.)

Format: DELETE filename$_1$ {...filename$_n$}

Purpose: To delete the files in a directory having names given in the argument list. A filename may be preceded by a directory specifier if the directory has been initialized. An attempt to delete a link will cause the link's resolution file, if any, to be deleted. To remove a link entry, use the UNLINK command.

Switches:

Global:

/C - confirm each deletion. Each filename is repeated, while the system waits for the operator to confirm that his file is to be deleted by typing a carriage return. To prevent the deletion from occurring, the operator types any key other than carriage return.

/L - list deleted files on $LPT (overrides /V).

/V - verify deletion with a list of names of deleted files on $TTO.

Local:

/A - delete only files created this date or after. Preceding argument is of the form MM-DD-YY where MM and DD can be one or two digits.

/B - delete only files created before this date. Preceding argument is of the form MM-DD-YY where MM and DD can be one or two digits.

/N - do not delete any files that match this name

Asterisk: Permitted only when filename argument is in the current directory.

Examples: DELETE LIMIT.-)

deletes all files having the name LIMIT and any extension (including null), e.g., LIMIT.SR, LIMIT.RB, LIMIT.SV, LIMIT..

DELETE/V -.LS)

deletes all files with the .LS extension, and lists their names:

DELETED A.LS
DELETED COM.LS
DELETED MAP.LS

DELETE A**B)

Name: DELETE (Continued)

Examples: (Continued)

deletes files AXYB, AXWB, but not AB or AXB.

DELETE A-B)

deletes files AZYB, AXWB, AB and AXB.

DELETE/C A-B.)

ask for confirmation of each file before deletion:

AZYB:)* File AZYB is deleted.
AXWB:# File AXWB is not deleted.

On confirmation if the user types) a "*" is echoed.
Any other character echos nothing.

Name: DUMP (Dump one or more files.)

Format: DUMP outputfilename { filename$_1$ ...} ↑
{ old filename/S new filename} . . .

Purpose: To dump a given file or files onto a given file or device. The directory information for each file--name, length, attributes, creation and last access time--is written as a header to each dumped file. If no file names are given, all non-permanent files are dumped. If file names are given, no name can be preceded by a device specifier. filename may be either a partition or subdirectory. Dumping a directory dumps the contents of the directory too.

If while dumping all files (DUMP/A) in an environment containing several directories you abort the dump by typing CTRL A, you should explicitly direct the CLI to a specific directory before making any further file references. This should be done since it is impossible to determine what directory was being dumped (and thus was the current directory) at the time you aborted the dump via CTRL A.

Switches:

Global:

/A - all files, permanent as well as non-permanent, are to be dumped.
/K - do not dump links
/L - list the dumped files on $LPT (overrides /V).
/S - dump a file on segments of paper tape. The file is punched in segments of up to 20K bytes each. Each segment of tape has a unique segment number placed in its dump heading, enabling the system to verify that tapes are later reLOADed in proper sequence.
/V - verify dump with a list of names of dumped files on the console.

Local:

/A - Dump only files created this day or later. Preceding argument is of the form MM-DD-YY where MM and DD can be one or two digits.
/B - Dump only files created before this day. Preceding argument is of the form MM-DD-YY where MM and DD can be one or two digits.
/N - Don't dump files that match this name.
/S - assign a new name to a file output in a dump (but retain its old name in the current directory).

Name: DUMP (Continued)

Asterisk: Permitted only when filename argument is in the current directory.

Examples: DUMP/A MT0:0 12-20-74/A

causes all permanent and non-permanent files created on Dec. 20, 1974 or after to be dumped onto file 0 of magnetic tape unit number 0. This tape file can then be saved as backup for the disk.

```
DUMP/L  DUMPFI  -.SV )
EDIT.SV  )
ASM.SV   }     list of files on $LPT
RLDR.SV  )
```

causes all non-permanent files with the extension SV to be dumped to the file DUMPFI and a list of files dumped to be output on the line printer.

```
DUMP/A  MT0:0   ABC/S   DEF )
```

File ABC will be named DEF in the dump (but its name remains ABC in this system) even if ABC is permanent. DEF will be dumped on file zero of MT0.

Name: ENDLOG (Close the log file.)

Format: ENDLOG { password }

Purpose: To close the log file which was opened by a previous CLI LOG command. If the previous LOG command used a password argument, this same argument must be used with the ENDLOG command.

This command, ENDLOG password, appears in the log file.

Switches: None.

Asterisk: Not permitted.

Example: ENDLOG GSTONE)

This command closes the current log file, permitting it to be TYPEd, PRINTed, etc. The password GSTONE is used since it was specified when the log file was last opened.

Name: FILCOM (Compare two files.)

Format: FILCOM filename$_1$ filename$_2$ { listing file/L }

Purpose: To compare two files, word by word, and print dissimilar word pairs. Dissimilar word pairs at the same displacements in the two files are printed on the console if the /L switch is not used Words are printed in octal and the displacement where they occur in the files is also printed. File organizations of the two files may differ.

Switches:

Local: /L - listing file

Asterisk: Not permitted.

Example: FILCOM YIN YANG $LPT/L)

causes a word-by-word comparison of the contents of files YIN and YANG. Any dissimilar word pairs will be printed in octal on the line printer, along with their respective word displacements in the files:

```
Ø25/  Ø44516  Ø4253Ø
141/  ØØØØ14  Ø2ØØ44
142/  ØØØØØØ  Ø4612Ø
143/  ØØØØØØ  Ø52Ø57
144/  ØØØØØØ  Ø46Ø15
145/  ØØØØØØ  ØØØØ14
```

If YANG were a null file, the entire contents of file YIN would be printed. Dashes would be printed for file YANG.

Name: FPRINT (Print a disk file in bytes, decimal, hexadecimal or octal.)

Format: FPRINT filename { filename/L }

Purpose: Print any readable disk file in either bytes, decimal, hexadecimal, or octal with any printable ASCII characters printed on the right side. Any non printing characters are reported as periods (.). The location counter is always in octal.

Switches:

Global:

/B	-	byte print out
/D	-	decimal print out
/H	-	hexadecimal print out
/L	-	line printer
/O	-	octal print out (default)
/Z	-	file starts at zero

Local:

/F	-	first location to be dumped
/L	-	overwrites global /L, directs output to file name that precedes it.
/T	-	last location to be dumped

Example: FPRINT/L TE1

causes file TE1 to be listed on the line printer. The mode is octal by default.

Name: LOAD (Reload dumped files.)

Format: LOAD inputfilename {filename$_1$ . . .}

Purpose: To load or list a previously dumped file or files from a given device or directory into the current directory. If no file names are specified, all files as specified by switches in the input file are loaded. The LOAD command can be used to list or load only those files which were previously DUMPed. Files to be loaded must bear names which are distinct from files existing in the current directory (unless /R or /N is given).

If a file to be loaded has the partition, directory, or link characteristics, these characteristics will persist. In the case of a partition, the partition will be re-created with necessary disk file space. It is possible to load files selectively from any directory which was DUMPED. Neither the DUMP nor the LOAD command changes file access/file creation information specified for the files.

If files were dumped in segments using the DUMP/S command, these files must be loaded in the same sequence that they were dumped in. Failure to follow the same sequence will result in a CLI runtime error message.

Switches:

Global:

- /A - all files, including permanent files.
- /B - brief listing
- /E - suppress non-fatal error messages.
- /I - ignore any checksum errors.
- /K - do not load links.
- /L - list loaded file names on the line printer. (Overrides /V switch and console listing by /N.)
- /N - only list the files, do not load them; output list on console. If global /R is used with /N, only the most recent version of the file will be listed.
- /R - load most recent version of file. The file's creation date is examined. If the existing file has the same or a more recent creation date than a file awaiting loading, the existing file is not deleted. If the current file's creation date is older than a file awaiting loading, the current file is deleted and the newer file is loaded.

Name: LOAD (Continued)

Switches:

Global: (Continued)

/V - verify the load with a list on the console of the names of files loaded. Subdirectory and secondary partition names will be indicated by being preceded and followed with line feeds.

Local:

/A - Load only files created this day or later. Preceding argument is of the form MM-DD-YY where MM and DD can be one or two digits.

/B - Load only files created before this day. Preceding argument is of the form MM-DD-YY where MM and DD can be one or two digits.

N - Don't load files that match this name.

Asterisk: Permitted only when the filename argument is in the current directory.

Examples: LOAD $PTR)

causes whatever previously dumped non-permanent files that are in the paper tape reader to be reconstructed on disk under the same names. File name, length, and attributes are entered in the file directory.

```
LOAD/V $PTR -.SV )
LOAD $PTR, STRIKE ANY KEY.
EDIT.SV
ASM.SV
```

causes loading of all files with the extension .SV and a list of the files loaded.

Name: LOG (Open the log file.)

Format: LOG {password} {directory/O}

Purpose: To record all CLI dialogue appearing on the console (a feature of particular value to users with video display consoles). CLI dialogue goes to a file named LOG.CM. Only one log file may exist at a time. This file may be deleted only after it is closed, and is closed only when an ENDLOG command is issued, the master directory is released, or the BOOT command is issued. If, for example, the system is RELEASEd, a new system is bootstrapped into execution, and the LOG command is reissued, the old LOG.CM will be appended to. This file cannot be examined, printed, etc., until it is closed. LOG.CM is updated after each line of CLI dialogue so that it will always be current.

The password is an optional argument, consisting of no more than ten decimal alphanumeric characters; this argument prevents the log file from being closed inadvertently. If a password is given, the same password must be used as an argument to the ENDLOG command in order to close LOG.CM.

The directory argument indicates a destination for the log file other than the current directory. This argument must be initialized before it is specified in the LOG command.

Switches:

Global: /H - Place a heading at the beginning of the log file. This heading consists of the title

LOG FILE

plus the following information: current directory name, master directory name and current system name (within brackets), and the current date and time.

Local: /O - output the log file to the preceding directory (the current directory is selected by default). The directory must be initialized.

Asterisk: Not permitted.

Name: LOG (Continued)

Example: LOG/H GSTONE)

This command causes all CLI dialogue, except for prompt messages, to be output to file LOG.CM. The heading at the beginning of this file is as follows:

```
***LOGFILE***GATE [DP0:SYS] 6/17/72  3:0:0
```

GATE is the name of the current directory.

Name: MKABS (Make an absolute file from a core image file.)

Format: MKABS save filename absolute binary filename

Purpose: To make an absolute binary file from a core image (save) file.

MKABS gives users the facility of converting files that are executable under the operating system into absolute binary files that can be executed without RDOS.

Switches:

Global:

/S - starting address switch. The starting address of the save file as specified in USTSA of the file will be used as the address for an absolute binary start block. Default is a null start block which causes the binary loader to halt when loading is completed.

/Z - save file begins with core location zero (by default, begins with location 16).

Local:

/F - first address (relative to location 0 of the save file) from which the absolute binary file is to be created.

/S - starting address switch. An absolute binary start block will be output with the starting address specified by the preceding octal argument.

/T - last address (relative to location 0 of the save file) which is to become part of the absolute binary file.

Asterisk: Not permitted.

Extensions: Search for save filename.SV. If not found, search for save filename.

Examples: MKABS FOO $PTP)

punches an absolute binary file on the paper tape punch from file FOO.SV or, if not found, from FOO.

MKABS FOO $PTP 1000/S)

punches an absolute binary file with a start block specifying 1000 as the starting address.

Name: MKSAVE (Make a save file from an absolute binary file.)

Format: MKSAVE absolute-binary-filename save-filename

Purpose: To create a core image (save) file from an absolute binary file.

Switches:

Global: /Z - create save file beginning at core location 0 rather than 16_8.

Local: None.

Asterisk: Not permitted.

Extensions: MKSAVE produces save-filename.SV as output, regardless of the extension specified by the save file argument.

Example: MKSAVE/Z $PTR DK0:A)

Causes creation of a core image file on fixed head disk unit 0 called A.SV, with the S attribute, from the absolute binary file loaded in the paper tape reader. Save file A.SV can now be executed via HIPBOOT see Appendix E of the RDOS User's Manual, 093-000075 or the BOOT command, provided the response to the "FILENAME?" query is "A.SV/A".

Name: MOVE (Move files from one directory to another.)

Format: MOVE destination directory name {filename$_1$ ...},
{old filename/S new filename} ...

Purpose: To move a given file or files in the current directory onto a given file or device. The directory information for each file--name, length, attributes, creation and last access time--is preserved. If no file names are given, all non-permanent files are moved. If file names are given, no name can be preceded by a device specifier. filename may be either a partition or subdirectory. Moving a directory moves the contents of the directory too.

Switches:

Global:

/A - all files, including permanent files.
/D - delete original file once transfer is complete.
/K - do not move links.
/L - list moved filenames on the line printer (overrides /V switch and console listing by /N).
/R - move most recent version of the file. The file's creation date is examined. If the file in the destination directory has the same or a more recent creation date than the file in the current directory the existing file is not moved. If the current directories files' creation date is older than the file awaiting a move, the current file is deleted and the newer file is moved.
/V - verify the move with a list on the console of the names of the moved files. Subdirectory and secondary partition names will be indicated by being preceded and followed with line feeds.

Local:

/A - move any file created this day or after, the preceding argument is of the form MM-DD-YY where MM and DD may be one or two digits.
/B - move any file created before this date. The preceding argument is of the form MM-DD-YY where MM and DD may be one or two digits.
/N - do not move any files that match this name.
/S - assign a new name to a moved file (but retain its old name in the current directory).

Asterisk: Permitted.

Example: MOVE/D/K MYDIR -.SR)

Name: MOVE (Continued)

Example: (Continued)

causes all non-permanent files with . SR extension to be moved into the destination directory MYDIR and delete the original files once the transfer is complete.

Name: PRINT (Print a file on the line printer.)

Format: PRINT filename$_1$ { filename$_2$ . . . }

Purpose: To print a given file or files on the line printer. The command is the equivalent of a series of XFER commands:

XFER / A filename$_1$ $LPT; . . . ;XFER / A filename$_n$ $LPT

The source files may come from any device. If a parity error is detected, a left slash (\ ) is printed in place of the bad character, and the message "PARITY ERROR" is output on the console; printing then continues.

Switches: None.

Asterisk: Not permitted.

Example: PRINT FOO.SR DP2:COM.SR EXT.SR $PTR)

causes source files FOO, COM, EXT, and one mounted on the high speed reader to be printed on the line printer.

PRINT RED:WHITE)

causes file WHITE in subdirectory RED to be printed on the line printer.

Name: PUNCH (Copy an ASCII file on the paper tape punch.)

Format: PUNCH filename$_1$ [filename$_2$ . . .]

Purpose: To copy a given file or files on the high speed punch. The command is the equivalent of a series of XFER commands:

XFER/A filename$_1$ $PTP; . . .;XFER/A filename$_n$ $PTP

The source files may come from any device. If a parity error is detected, a left slash (\) is punched in place of the bad character, and the message "PARITY ERROR" is output; punching continues.

Switches: None.

Asterisk: Not permitted.

Examples: PUNCH DK0:ALPHA.SR BETA.SR $TTR)

causes files ALPHA.SR, BETA.SR, and one mounted on the teletype reader to be punched on the high speed punch.

Name: RELEASE (Release a device from the system.)

Format: RELEASE device-specifier
RELEASE directory

Purpose: To prevent further I/O access to a directory or a device, or to rewind a magnetic tape unit. The command should be issued before any disk pack is physically removed from a removable disk unit. No further access to the disk device, mag tape, cassette, or directory is then permitted unless an INIT or DIR command is executed. If a directory other than a disk global specifier was the master directory, this directory must be released if the system is to be shut down. A release of the master directory causes all initialized devices in the system (including tape transports) to be released. A release of any primary partition causes all directories within the partition to be released.

If in a background-only environment the master disk device is released, the message "MASTER DEVICE RELEASED" is output on the console, and all core memory within the program area is set to DOC 0, CPU or HALT (063077 octal). A system bootstrap may be performed immediately after a master device is released, without the need to cycle the disk drive through its load sequence. In a dual program environment, the master device cannot be released from the background.

The default directory device for the foreground is initially the same as the default directory device for the background. All files in a directory must be closed before the directory can be released.

Switches: None.

Asterisk: Not permitted.

Example: RELEASE DP1)

This command releases primary partition DP1 and all directories in this partition, and permits the disk pack to be removed from moving head disk unit 1.

RELEASE MT0)

MT0 will be rewound.

Name: RENAME (Change the file name.)

Format: RENAME oldname$_1$ newname$_1$ [. . . oldname$_n$ newname$_n$]

Purpose: To change the current name of a file or files.

Switches: None.

Asterisk: Not permitted.

Example:

```
DELETE  Q.SV )
R
RENAME  QTEST.SV  Q.SV )
R
```

The above commands replace the old version of Q.SV with a new version, one previously named QTEST.SV.

```
RENAME  DK0:A1   DK0:A   B1   B )
```

Rename file A1 to A on fixed head disk unit 0. Rename file B1 to B on the default directory.

Name: REV (Display the version level of a save file.)

Format: REV filename[.SV]

Purpose: To display the revision level of a save file. The save file must have the "S" attribute. Revision level information will be displayed as a major revision number followed by a period and a minor revision number. Both major and minor revision levels can be in the range 0-99 inclusive.

The .REV pseudo-op is used to assign a major and minor revision level number to a save file. If this pseudo-op is not used in a save file, then the revision levels of this save file will be displayed as "00.00".

An RDOS system save file cannot be supplied as an argument to the REV command even if the save file is CHATRed "S". The response to such a REV command would not be meaningful.

Switches: None.

Asterisk: Not permitted.

Example: REV TEST.SV)

This command causes a response of "03.07" to be returned by the CLI. This response indicates that the major revision level of TEST.SV is 03, and the minor revision level of this file is 07.

Name: SAVE (Save the core image as a save file.)

Format: SAVE filename

Purpose: To create a save file from the file named BREAK.SV on the default directory device. SAVE is commonly used to save the core image of a program interrupted by a CTRL C break or by the debugger $V command. SAVE causes the most recent core image saved under the name BREAK.SV (FBREAK.SV for the foreground) to be given a new name by deleting filename.SV (if it exists) and renaming the BREAK.SV (or FBREAK.SV) to filename.SV. filename cannot be preceded by a device specifier.

Switches: None.

Asterisk: Not permitted.

Extensions: Output always has the SV extension. If the filename argument already has an extension, the extension will be ignored, e.g. either

SAVE GAMMA) or SAVE GAMMA.YY)

would produce the save file GAMMA.SV.

Example:

```
DEB ALPHA )          ←enter debugger to correct location PP
PP/ LDA 2 @0 LDA 2 @0 3
$V )
BREAK                ←exit from debugger
SAVE ALPHA )         ←save core image as ALPHA.SV
```

Name: TYPE (Output file contents on the system console.)

Format: TYPE filename$_1$ {filename$_2$ . . .}

Purpose: To copy a given file or files on the program console. The command is the equivalent of a series of XFER commands:

XFER/A filename$_1$ $TTO;. . .;XFER/A filename$_n$ $TTO)

The source files may come from any device. When a character with bad parity is detected, a left slash (\) is typed and the message "PARITY ERROR." is output; typing of the remainder of the file continues.

Switches: None.

Asterisk: Not permitted.

Example: TYPE A.SR B.SR $PTR DP1:XX.SR)

This command causes the following files to be typed or displayed on the program console: Disk files A.SR and B.SR, a file mounted in the high speed reader, and source file XX in primary partition DP1.

Name: UNLINK (Delete a link entry name.)

Format: UNLINK linkname$_1$ [. . . linkname$_n$]

Purpose: To remove one or more link entries.

Switches:

Global:

/C - confirm each removal. Each link entry name is repeated, while the system waits for the operation to confirm that this link entry is to be deleted by typing a carriage return. To prevent the deletion or unlinking from occurring, the operator types any key other than carriage return.

/L - list the removed files on $LPT (overrides /V).

/V - verify the removal with a list of names of unlinked files.

Local: None.

Asterisk: Permitted only when the filename argument is in the current directory.

Example: UNLINK/C TEST.-)

This command asks for a confirmation before each link entry is removed:

TEST.SR)*	Link TEST.SR is removed
TEST.RB)*	Link TEST.RB is removed
TEST.SV %	Link TEST.SV is not removed

On confirmation if the user enters) a "*" is echoed. Any other character echo's nothing.

Name: XFER (Copy the contents of a file into another file.)

Format: XFER sourcefile destinationfile

Purpose: To transfer a file to another file, organizing the destination file differently if desired. If the destination file does not exist, it is created. By default the destination file is organized sequentially.

The system will detect parity errors in ASCII files, and it will detect parity errors in binary files on magnetic or cassette tape. When a parity error is detected in the input file, the message PARITY ERROR, FILE: xxx will be output to the console. If one or more parity errors are detected in a paper tape file, a back-slash will be substituted for each bad character. If a parity error is detected in a magnetic tape or cassette file, the transfer is terminated.

Switches:

Global: By default, files are transferred sequentially without alteration. There are two switches:

- /A - ASCII transfer. Transfer the file line by line taking appropriate read/write action, such as inserting line feeds after each carriage return when transfer is from disk to line printer.
- /B - append the source file to the end of the destination file.

Local: By default, the destination file will be organized sequentially. The source file may be organized either sequentially, randomly, or contiguously.

- /R - organize the destination file randomly. (/R is not used on the source file.)
- /C - organize the destination file contiguously. (/C is not used on the source file, and may be used only if the source file is a disk file.)

Asterisk: Not permitted.

Name: XFER (Continued)

Examples:

```
XFER $PTR Q )
```

causes the file in the paper tape reader to be transferred to a disk file named Q.

```
XFER/A ALPHA.SR $LPT )
```

causes ALPHA.SR to be printed on the line printer.

```
XFER $PTR $PTP )
```

causes another tape to be punched, identical to the one read from the paper tape reader.

```
XFER DP0:MYFILE DP1:MYFILE )
```

transfers MYFILE from disk pack unit 0 to disk pack unit 1.

```
XFER A B/R )
```

causes file A to be copied, with random file organization, onto a file named B.

Name: CDIR (Create a subdirectory.)

Format: CDIR name

Purpose: To create a subdirectory with a .DR extension.

Switches: None.

Asterisk: Not permitted.

Examples:

```
DIR DP0:ALEPH )
CDIR BETH )
```

This pair of commands creates subdirectory BETH.DR in secondary partition ALEPH of DP0. CDIR DP0:ALEPH:BETH is an equivalent command.

Name: CPART (Create a secondary partition.)

Format: CPART name blockcount

Purpose: To create a secondary partition. The secondary partition is given the name specified in name with a .DR extension, and is a contiguous disk file whose length is specified (in decimal) in blockcount. blockcount must be 48 decimal or greater, and is a multiple of 16 decimal. If blockcount is not an integer multiple of 16, the system truncates it to the next lower multiple.

Switches: None.

Asterisk: Not permitted.

Example:
```
DIR DP0 )
CPART ALEPH 48 )
```

This pair of commands creates secondary partition ALEPH in primary partition DP0. ALEPH is 48 blocks in length. An equivalent command would be CPART DP0:ALEPH 48).

Name: DIR (Change the current default directory.)

Format: (1) DIR primary partition {:secondary partition} {:subdirectory}

(2) DIR {primary partition:} { {secondary partition} / {secondary partition:} subdirectory }

Purpose: To change the current directory or default directory device. At bootstrap time, a master device is established as the current directory. The DIR command permits another device or directory to be substituted as the default device/directory. If necessary, this command will also initialize the device/directory.

If the primary partition is not specified (second format), then the current primary partition is implied. Moreover, if in this case a subdirectory is specified, then it must be contained as an entry within the current primary partition.

Switches: None.

Asterisk: Not permitted.

Examples: DIR DP0)

Change all default file name references to the moving head disk, unit number 0.

DIR DP1:DEF:GHI)

Direct all file name references to subdirectory GHI in secondary partition DEF found in primary partition DP1.

Name: EQUIV (Assign a new name to a directory specifier.)

Format: EQUIV new name oldname

Purpose: To assign a new name to a multiple file device before the device is initialized. The old name will usually be a default global specifier, e.g., CTn, CT1n, DK0, DK1, DPn, MTn or MT1n where n is an integer from 0 to 7. However, the old name may be a previous new name, provided the device has not since been initialized.

The EQUIV command cannot be issued for a master disk device, nor can it be issued for a device after it has been initialized. After being released, a device's default global specifier is re-assigned to it. Subdirectories and secondary partitions cannot be EQUIValenced.

Switches: None.

Asterisk: Not permitted.

Example: Throughout an assembly language program, all magnetic tape file references have been made to TAPE. This provides device independence to the program at run time. Prior to the running of this program, the command

EQUIV TAPE MT0)

is issued. Now, all file references to TAPE, e.g., TAPE:2, will be resolved as references to magnetic tape unit zero, e.g., MT0:2.

Name: GDIR (Print the current directory device name.)

Format: GDIR

Purpose: Get the name of the current directory.

Switches: None.

Asterisk: Not permitted.

Example:
```
GDIR )
MANHATTAN
```

MANHATTAN is the current directory to which all file references are directed by default.

Name: INIT (Initialize a directory or device.)

Format:

(1) INIT device-specifier

(2) INIT primary partition [:secondary partition] [:subdirectory]

(3) INIT [primary partition:] { secondary partition / [secondary partition:] subdirectory }

Purpose: To initialize a magnetic tape or cassette unit, or a directory without making it the current directory. Files in non-current directories can be opened, deleted, etc. by means of the colon convention (e.g., DP0:AB) only if those directories are first initialized. Until the directory or device is released (RELEASE command) all files on the initialized tape unit, in the directory, or in the primary partition are now available to the user.

If primary partition is not specified (third format), then the current primary partition is implied. Moreover, if in this case a subdirectory is specified, then it must be contained as an entry within the current primary partition.

When a directory is initialized, entries for all peripherals supported by the operating system are made in that directory's SYS.DR. Since one of several differently configured operating systems may be in execution at any time, SYS.DR may grow in size when a directory is initialized, depending upon the peripheral support of the currently executing operating system. Thus if a disk partition is created with a minimum size of 32_{10} disk blocks, the partition may be initializable under one version of RDOS but not under a different configuration of the system.

Switches:

Global: By default, when a disk is initialized, the primary partition directory of the device is found and read into the system, allowing access to all files in the primary partition.

/F - Full initialization. Clears all previous files and information from a specified disk device and writes a new file directory and free storage map on the device. Full initialization of a non-master device causes blocks 0-5 of the master device to be written to the non-master device. (These blocks are reserved for HIPBOOT.) Full initialization of a tape device causes the tape to be rewound and two EOF's to be written, effectively erasing the tape.

Local: None

Aterisk: Not permitted.

Name: INIT (Continued)

Examples: INIT DP3)

Initialize the disk pack on unit number 3.

INIT/F MT1)

Magnetic tape or cassette initialization causes the tape to be rewound. Full (/F switch) initilization of MT1 causes the tape on drive MT1 to be rewound and two EOF's written on the tape, effectively erasing the tape.

INIT DP1:ABC

All files in secondary partition (or subdirectory) ABC will now be accessible, provided either explicit DIR commands are issued or file accesses are directed by means of explicit global/directory specifiers with colon separators.

Name: LINK (Create a link to a file on the same or another directory.)

Format: LINK filename/2

LINK link filename { directory specifier: } filename

Purpose: To create a link entry to another link or to a resolution file. If the file names are the same (as when the local/2 switch is given), the resolution file is presumed to exist in the current primary partition (i.e., the primary partition of the link entry's residence). The current or default directory is always presumed unless an alternate directory is specified.

If a link file name is given which differs from the resolution filename, the link file name is an alias. A resolution file may exist either in the current primary partition or, if directory specification information precedes the filename, in some other directory or partition.

Switches: None (except general local switches, like/2).

Asterisk: Not permitted.

Examples: LINK ASM.SV/2)

This command causes a link entry named ASM.SV to be made in the current directory to a file named ASM.SV in the current primary partition. That is, if the current directory is found on DP3, the resolution filename ASM.SV will be found in the primary partition of DP3. Note that the creation of a link in no way implies that the resolution file exists; thus links may be made to non-existent files. The detection of an unresolved link will be made only when an attempt is made to reference the resolution file, at which time the error message "FILE DOES NOT EXIST" will be given.

LINK A.SV O.SV)

This command causes a link entry named A.SV to be made in the current directory to a save file named O.SV in the current primary partition.

Name: LINK (Continued)

Examples: (Continued)

```
LINK    ASM.SV  DP2:OASM.SV )
```

This command causes a link entry named ASM.SV to be made in the current directory to an alias file named OASM.SV in primary partition DP2.

```
LINK    ASM.SV  SAM:ASM.SV )
```

This command causes a link entry named ASM.SV to be made in the current directory to a file of the same name in secondary partition (or subdirectory) SAM.

Name: LIST (List file directory information.)

Format: LIST { filename } . . . }

LIST { primary partition: } { secondary partition: } ↑
{ subdirectory: } filename

Purpose: To list information from the default directory or from any other directory about one or more files or link entries. If LIST has no filename argument, all non-permanent files and link entries in the current directory are listed.

Information which is listed for the files includes the file name and one or more of the following: file size (in bytes), resolution file attributes, link access attributes, file creation date and time, date last opened, file starting address (UFTAD of UFD), and decimal file use count. The resolution file attributes and link access attributes--if any--are separated by a right slash. The file starting address is enclosed within brackets, and the file use count is terminated with a period to indicate that it is a decimal figure.

The following information is listed for link entries: link entry name, and resolution entry name with directory specification name that was given when the link was created. An at-sign (@) is printed when the resolution file was defined to exist in the current primary partition.

The following is a list of file attributes and their meanings:

Printed Code		Meaning
P	-	permanent file, which cannot be deleted or renamed.
S	-	save file (core image).
W	-	write-protected file, which cannot be written.
R	-	read-protected file, which cannot be read.
A	-	attribute-protected file. The attributes of such a file cannot be changed. After the A attribute has been set it cannot be removed.
N	-	no resolution allowed. This attribute prevents a file from being linked to.
?	-	first user-definable attribute (bit 9).
&	-	second user-definable attribute (bit 10).

Name: LIST (Continued)

The following is a list of file characteristic codes and their meanings:

Printed Code		Meaning
D	-	random file organization
C	-	contiguous file organization.
I	-	accessible by direct I/O only. (Only SYS.DR and MAP.DR have this attribute).
L	-	link entry. Properly speaking, the L characteristic is given to directory entries rather than to the files themselves.
T	-	partition. This defines a file as being a partition.
Y	-	directory. This characteristic defines a file as being a directory.

The following is a list of link access characteristic codes and their meaning:

Printed Code		Meaning
/P	-	permanent file, which cannot be deleted or renamed.
/S	-	save file (core image).
/W	-	write-protected file, which cannot be written.
/R	-	read-protected file, which cannot be read.
/A	-	attribute-protected file. The attributes of such a file cannot be changed. After the A attribute has been set it cannot be removed.
/N	-	no resolution allowed. This attribute prevents a file from being linked to.
/?	-	first user-definable attribute (bit 9).
/&	-	second user-definable attribute (bit 10).

Switches:

Global:

/A	-	list all files within the current directory, both permanent and non-permanent files, giving file name, byte count, file attributes, and file characteristics.
/B	-	brief listing; gives only the file name.
/C	-	list the creation time (year/month/day hour:minute).
/E	-	list every category of file information (overrides /B, /C, /F, /O, and /U switches).

Name: LIST (Continued)

Switches: (Continued)

Global: (Continued)

/F - list the first address, i.e., the logical address of the first block in the file; list 0 if unassigned. (Enclosed within brackets.)

/L - output listing on line printer.

/K - no links

/O - list date file last opened (month/day/year).

/S - sort the output list alphabetically.

/U - list file use count (in decimal terminated with a period).

Local:

/A - list files created this date or after. The preceding argument is of the form MM-DD-YY where MM and DD can be one or two digits.

/B - list files created before this date. The preceding argument is of the form MM-DD-YY where MM and DD can be one or two digits.

/N - do not list any files that match this name.

Asterisk: Permitted only when the filename argument is in the current directory.

Example: LIST/E/A)

This command causes all information for all files and link entries in the current directory to be output on the console. A typical line of information describing a file would look like the following:

```
FLI.SV  8160  SD  03/23/73  13:56  03/23/73  [004164]  0
```

In this example, FLI.SV is the file name, consists of 8,160 bytes, is a randomly organized save file, was created on 1:56 p.m. of the 23rd day of March, 1973, was last accessed on that same date, has a starting logical block address of 4164, and has a file use count of zero.

Typical lines describing link entries would look like the following:

```
ABC.SV      DP0:DEF.SV
XXX.SV      @:XXX.SV
```

In the first link example, the link entry name is ABC.SV, zero byte length is shown and only the link characteristic is indicated since the entry is a link; the resolution file was defined to exist with alias DEF.SV in primary partition DP0. In the second example, the link entry name is XXX.SV and the resolution file was defined to have the same name and to reside in the current primary partition. Link entries are always indicated by having zero byte length and only the link characteristic.

Name: MDIR (Get the name of the current master directory.)

Format: MDIR

Purpose: To type the name of the current master directory on $TTO. This is the directory which contains the system save and overlay files, the spool file and push space for program swaps.

Switches: None.

Asterisk: Not permitted.

Example: MDIR)

This command causes the following master directory name to be typed on $TTO:

DP0

FOREGROUND/BACKGROUND COMMANDS

This section includes the CLI commands that are available for modifying foreground/background area. The following list indicates the commands as they are presented.

EXFG	-	Execute a program in the foreground.
GMEM	-	Get the foreground/background memory size in a mapped system.
SMEM	-	Set the foreground/background memory size in a mapped system.

Name: EXFG (Execute in the foreground.)

Format: EXFG filename
EXFG program-development-command stream

Purpose: To load into core memory a foreground save file from the background and transfer control either to the save file or to the debugger. Alternatively, to issue a utility command stream from the background which will be executed in the foreground. A foreground communications file is created for a utility command stream with a structure similar to COM.CM. This file is named FCOM.CM. If the save file cannot be loaded without overwriteing the CLI, the foreground save file will not be loaded.

Switches:

Global:
/D - control goes to the debugger.
/E - foreground and background have equal priority.

Asterisk: Not permitted.

Example: EXFG DP1:RMON)

causes the save file RMON in primary partition DP1 to be loaded into core if this is possible without overwriting the CLI. (In mapped systems the foreground program will never overwrite the CLI.) If in an unmapped system the loading of RMON would overwrite the CLI, the error message:

INSUFFICIENT MEMORY TO EXECUTE PROGRAM.

is output and RMON is not loaded. If, however, the loading of RMON is successful, the user may attempt to load and execute other background programs (which would overwrite the CLI). If there is insufficient background memory available to load a background program, the "insufficient memory" error message is output and the CLI remains active.

EXFG ASM FILEA $LPT/L)

This command causes FILEA to be assembled in the foreground, with an assembler listing output on the line printer.

NOTE: The program-development-command stream argument is acceptable in unmapped systems only when the utility save file (MAC, RLDR, etc.) has been previously loaded with partition information.

Name: GMEM (Get the foreground/background memory size in a mapped system.)

Format: GMEM

Purpose: To determine the current size of memory allocations for the foreground and background program areas. The size is given in 1024_{10} word blocks. This command may be used on machines with mapped addressing only, and is meaningful only when both a foreground and a background program exist.

Switches: None.

Asterisk: Not permitted.

Example:
```
GMEM )
BG:  19  FG:  18
```

19,456 words of memory (19 times 1024) are available to the background, and 18,432 words of memory are available to the foreground.

Name: SMEM (Set the foreground/background memory size in a mapped system.)

Format: SMEM background foreground

Purpose: To re-allocate the amounts of user memory which will be available to the background and foreground programs. Memory is allocated equally to the foreground and background by default. The background and foreground parameters used in this command are decimal numbers indicating the number of 1024 word blocks which are to be allocated to each program area.

This command must be issued in mapped systems only, from the background CLI, while no foreground program is running.

Switches: None.

Asterisk: Not permitted.

Example: SMEM 8 12)

This command allocates eight 1024-word blocks of memory for the background and twelve 1024-word blocks for the foreground.

SYSTEM GENERATION COMMANDS

This section includes the CLI commands applicable for system generation. The following list indicates the commands as they are presented.

SYSGEN	-	Generate a new system
SQUASH	-	Prepare a system save file for use in bootstrapping.

Name: SYSGEN (Generate a new system.)

Format:
(1) SYSGEN { filename/V } { listing file/L }
(2) SYSGEN { filename/A/S } { listing file/L }

Purpose: To generate a new RDOS system and optionally to perform the system load without the need for further operator intervention. This command will cause the SYSGEN dialogue to be output on $TTO as discussed in Appendix E of the RDOS User's Manual, 093-000075. Following the last SYSGEN interrogation, if global /N is specified, system loading will not be performed automatically; the indirect command files (SRLDR.CM etc.) need not be disk resident when this command is issued. See Appendix E of the RDOS User's Manual, 093-000075, for operations to be performed after the system loading. Load errors will be output on $TTO.

The SYSGEN dialogue can be saved in a user-specified file by means of the /V local switch (format 1) even while that dialogue is being used to generate a system. The file containing the dialogue can then later be used directly (format 2) to generate a system meeting the same specifications.

Switches:

Global:
/N - do not perform automatic system load.

Local:
/A - a dummy switch applied to filename when /S is also used and the user wishes to exercise system load without operator intervention and generate a system using the SYSGEN dialogue file.
/L - Output load map to the specified listing file. (Default is TTO.)
/S - Generate save file of a mapped system or squash and rename SYS 000.SV and rename it to a specified filename in an unmapped system.
/T - Search for tuning file (filename or filename.TU) and enter in it user responses to override previously recorded SYSGEN responses.
/V - Save the SYSGEN dialogue for the system being generated in a specified file (filename).

Example: SYSGEN

This command causes the system generation program to be activated so that the operator may tailor a new RDOS system by

Name: SYSGEN (Continued)

Example: (Continued)

answering a series of questions. Upon answering the last question SYSGEN will invoke the relocatable loader to produce an executable version of RDOS. Load errors and the load map go to the $TTO.

Note: Execution of this command causes residual files, SYSGEN.CM and CLI.CM to remain on disk. These files can be deleted by the user, if desired.

Name: SQUASH (Prepare a system save file for use in bootstrapping.)

Format: SQUASH sourcefile destinationfile

Purpose: To produce a system save file in squashed format and to create an overlay file if needed. In squashed format, no disk space is used to store images of user address space.

The sourcefile name is the name of the save file which is to be squashed; an overlay file of the same name exists. (If the /N switch is used, a destination overlay file is presumed to exist.) The destinationfile is the name to be given to the executable system save and overlay files. If the destination files already exist (but are not permanent), they will be deleted and new save and overlay files will be created.

Switches:

Global: /N - Do not create a destination overlay file since one already exists.

Asterisk: Not permitted.

Extensions: The .SV extension is implied by both the source and destination file names.

Example: SQUASH SYS1 JOE)

This command causes SYS1.SV to be squashed and named JOE.SV, and causes SYS1.OL to be named JOE.OL.

TUNING FILE COMMANDS

This section includes CLI commands that are applicable to the tuning file. The following list indicates the commands as they are presented.

TUON	-	Initiate recording in the tuning file.
TUOFF	-	Stop recording in the tuning file.
TPRINT	-	Print the tuning file.

Name: TUON (Initiate recording in the tuning file)

Format: TUON

Purpose: Initiate recording in the tuning file the following system facilities:

- buffer number - number of requests and number of failures.
- stack number - number of requests and number of failures.
- number of system overlay requests - number of requests and number of failures.

Additionally, an overlay frequency report is compiled.

Switches: None.

Asterisk: Not permitted.

Example: TUON)

initiates recording in the tuning file.

Name: TUOFF (Stop recording in the tuning file)

Format: TUOFF

Purpose: To stop the recording of system facilities in the tuning file. TUOFF does not delete the contents of the tuning file.

Switches: None.

Asterisk: Not permitted.

Example: TUOFF)

terminates recording in the tuning file.

Name: TPRINT (Print the tuning file)

Format: TPRINT filename

Purpose: Initiate the printing of the tuning file which contains the following system facilities:

- buffer number - number of requests and number of failures.
- stack number - number of requests and number of failures.
- number of system overlay requests - number of requests and number of failures.

filename is the name of the system tuning file that is to be reported. It has the same name as the system that the tuning was monitored under.

Additionally, an overlay frequency report can be printed.

Switches:

Global:
/L - print on the line printer.
/O - print overlay frequency report.

Asterisk: Not permitted.

Example: TPRINT/L BSYS)

causes the tuning file for the system BSYS to be printed on the line printer.

SPOOLING COMMANDS

This section includes CLI commands applicable to the spooling facility. The following list indicates the commands as they are presented.

SPDIS	-	Disable device spooling.
SPEBL	-	Enable device spooling.
SPKILL	-	Stop a spooling operation.

Name: SPDIS (Disable device spooling.)

Format: SPDIS device-name$_1$ [device-name$_2$ . . .]

Purpose: To disable spooling on a device. The device-names may be the names of any user devices defined as spoolable or any of the following:

$DPO, $LPT, $LPT1, $PLT, $PLT1, $PTP, $PTP1, $TTO, $TTO1, $TTP, $TTP1.

Switches: None

Asterisk: Not permitted.

Example: SPDIS $LPT)

This command prevents data output to the line printer from being spooled. To reinstitute spooling, the command SPEBL $LPT must be issued. If output is currently being spooled to the line printer, the disablement of spooling will become effective after the current spool is completed.

Name: SPEBL (Enable device spooling.)

Format: SPEBL device-name$_1$ { device-name$_2$ . . . }

Purpose: To enable spooling on a spoolable device. The device-names may be the names of any user devices defined as spoolable or any of the following:

$DPO, $LPT, $LPT1, $PLT, $PLT1, $PTP, $PTP1, $TTO, $TTO1, $TTP, $TTP1

Switches: None.

Asterisk: Not permitted.

Example: SPEBL $LPT)

This command causes data output to the line printer to be spooled.

Name: SPKILL (Stop a spooling operation.)

Format: SPKILL device-name$_1$ { device-name$_2$ . . . }

Purpose: To stop a spool operation. The device-names may be the names of any user devices defined as spoolable or any of the following:

$DPO, $LPT, $LPT1, $PLT, $PLT1, $PTP, $PTP1, $TTO, $TTO1, $TTP, $TTP1

Switches: None.

Asterisk: Not permitted.

Example: SPKILL $LPT)

This command stops the spooling of data to a line printer, causing it to be output in unbuffered fashion (and hence making its output terminated by CTRL A, if desired).

TIME, DATE, AND SYSTEM NAME COMMANDS

This section includes the CLI commands applicable for getting and setting the time and date as well as the name of the current operating system. The following list indicates the commands as they are presented.

GSYS	-	Get the name of the current operating system.
GTOD	-	Get the time and the date.
SDAY	-	Set today's date.
STOD	-	Set the time.

Name: GSYS (Get the name of the current operating system.)

Format: GSYS

Purpose: To type the name of the current operating system on $TTO.

Switches: None.

Asterisk: Not permitted.

Example: GSYS)

This command causes the following operating system name to be typed on $TTO:

SYS

Note that the save file extension, .SV, is not typed.

Name: GTOD (Get the time and date.)

Format: GTOD

Purpose: To get the time of day and the current date.

Switches: None.

Asterisk: Not permitted.

Example:

```
GTOD )
02/17/73    21:24:20
R
```

The message indicates that the time is 20 seconds after 9:24 p.m., and the date is February 17, 1973.

Name: SDAY (Set today's date.)

Format: SDAY month day year

Purpose: To set the system calendar. The year information may be either two or four digits (e.g., 73 or 1973). Only spaces or commas can be used to separate the date arguments.

Switches: None.

Asterisk: Not permitted.

Example: SDAY 4 17 1973)

sets the system calendar to April 17, 1973.

Name: STOD (Set the time.)

Format: STOD hour minute second

Purpose: To set the system clock. (The system clock is a 24 hour clock.) Only spaces or commas can be used to separate the time arguments.

Switches: None.

Asterisk: Not permitted.

Example: STOD 21 24 0)

causes the system clock to be set to 9:24:0 p. m.

BOOTSTRAP COMMAND

This section includes the CLI command applicable to perform a disk bootstrap.

BOOT - Perform a disk bootstrap.

Name: BOOT (Perform a disk bootstrap.)

Format:

(1) BOOT primary partition

(2) BOOT [primary partition:] secondary partition

(3) BOOT [primary partition:] [secondary partition:] save file name

Purpose: To release the current system and perform a disk bootstrap. The disk bootstrap program, HIPBOOT, must reside on the disk device named in primary partition or on the disk unit containing the save file which will be executed. All partition arguments must be initialized before the BOOT command is issued. No save file can be bootstrapped in a subdirectory.

Use of a device specifier alone (the first format) activates HIPBOOT on that device; thus a file name response will be required when HIPBOOT gains control. The primary partition must be one of the following: DK0, DK1, DP0...DP7. EQUIVed primary partition names may not be used.

If in the second or third formats the colon convention is used, no space separators may occur between a colon and its argument. If in the third format no save file name is specified, the default operating system, SYS.SV/SYS.OL, is indicated. If save file name is specified, it must have no extension other than ".SV". Additionally, save file name may be a link, the name of an RDOS system, or the name of a save file (with certain restrictions outlined in Appendix E, "Disk Bootstrapping with Operator Intervention" of the RDOS User's Manual, 093-000075). If the operating system being run is a link, the directory containing the resolution entry becomes the master directory; moreover, that system's overlay file must also be linked.

If all front panel switches are up when the BOOT command is given, any save file name given will be ignored and the default system, SYS.SV/SYS.OL, will be bootstrapped. Additionally, the system will attempt to chain to a save file named RESTART.SV (as in power fail-auto restart applications). If this feature is not desired, ensure that the front panel switches are not all in the up position.

Note that there is no switch available in the BOOT command. Thus if you wish to bootstrap an absolute program which does not conform to the necessary conventions described for use with HIPBOOT, you must activate HIPBOOT by using the first format, then use the /A switch in response to the file name query output by HIPBOOT. For more information describing the use and operation of HIPBOOT, see Appendix E of the RDOS User's Manual, 093-000075.

Name: BOOT (Continued)

Switches: None.

Asterisk: Not permitted.

Examples: BOOT DP0)

This command loads HIPBOOT from moving head disk unit 0. Upon being loaded, HIPBOOT outputs the query: FILENAME ?. A system file name response is then given, as described in Appendix E of the RDOS User's Manual, 093-000075.

BOOT DP1:TEST:SYS32K)

This command loads RDOS system SYS32K found in secondary partition TEST of primary partition DP1. TEST and DP1 must have been initialized prior to the BOOT command. When SYS32K.SV is loaded, time and date information will be requested, after which the operating system's CLI will be active.

BOOT DP0:RTOS

This command loads the RTOS save file found in primary partition DP0. No time and date information will be requested when RTOS.SV is loaded. When the execution of RTOS.SV is completed, an RDOS system must be bootstrapped via the front panel switches unless the RTOS program itself activates HIPBOOT.

BOOT DP0
FILENAME ? DP0:SPART:FOO.SV/A

This command and response to the HIPBOOT filename query cause absolute program FOO.SV to be executed. FOO.SV resides in secondary partition SPART, and does not conform to the conventions required by HIPBOOT (thus the use of /A).

CHAIN COMMAND

This section includes the CLI command used to overwrite the CLI with another program.

CHAIN - Overwrite the CLI with another program.

Name: CHAIN (Overwrite the CLI with another program.)

Format: CHAIN save file name

Purpose: To overwrite the CLI by performing a program chain to save file name. This command should be issued only in special circumstances and with caution. If this command is issued from level zero, the operating system must be rebootstrapped before other CLI commands can be issued, unless the user program chains control back to the CLI via .EXEC.

Switches:

Global: /D - control goes to the debugger.

Asterisk: Not allowed.

Example: CHAIN DEMO)

Suppose DEMO.SV is a demonstration program on a dual Nova system which performs some specified operation, and then returns control back to the CLI at level 0. Suppose further that the system is designed to be fail-safe; if power should fail in the first processor, N1, control will be gained in the second processor, N2, via the IPB. A start-up program in N2 (RESTART.SV) restores the status of parameters saved in a common disk file and then chains to the save file DEMO.SV. When the operation of DEMO.SV is complete in N2, the program chains to the CLI; a .RTN to the next higher level would be impossible, since RESTART.SV existed at level 0. Since DEMO.SV must be capable of returning to the CLI at level 0 in both N1 and N2, DEMO.SV must originally be CHAINed into operation in N1.

DISK BLOCKS AVAILABLE/USED COMMAND

This section includes the CLI command available to determine the amount of disk used and available.

DISK - List the number of disk blocks available and used.

Name: DISK (List the number of disk blocks used and remaining.)

Format: DISK

Purpose: To obtain a count in decimal of the number of blocks used and the number of blocks left for the user in the current partition. If the current directory is a subdirectory, the size of the parent partition is indicated. The net number of blocks available to a user (n) is always less than the number of physical blocks (p) on the disk, due to the space requirements for HIPBOOT and the structure of map directories under RDOS. The value n reported by the DISK command is the largest integer multiple of 16_{10} which is less than or equal to p-6.

Switches: None.

Asterisk: Not permitted.

Example:
```
DISK )
LEFT:  2734   USED: 2130
```

The response indicates that 2734 out of a total of 4864 blocks are still available for use. Thus this response was given in the primary partition of a 4047A moving head disk (with 4872 physical blocks).

COMMUNICATION COMMAND

This section includes the CLI command for transmitting a program via a MCA line.

MCABOOT - Transmit a program via a MCA line.

Name: MCABOOT (Transmit a program via a MCA line.)

Format:

```
MCABOOT/F { MCAT1:n } [ [ { system name/S } ] ] [ filename... ]
          { MCAT:n  }   [ [ { user program/S } ] ]

MCABOOT   { MCAT1:n } [ [ { system name/S } ] ]
          { MCAT:n  }   [ [ { user program/S } ] ]
```

Purpose: To transmit an operating system or program executable by HIPBOOT from one CPU to another via a multiprocessor communications adapter. The transmitter, MCAT or MCAT1, must be part of the same network as the receiver, n. An operator at the receiving CPU must have requested the sender's transmission by first placing 100007 or 100047 in the receiver's data switches, then by depressing "RESET", followed by "PROGRAM LOAD". (In systems lacking "PROGRAM LOAD", deposit 377 in location 377, deposit 60107 or 60147 in 376, and "START" at 376.) The transmitting unit will wait for the receiver to request the transmission, up to the default timeout-period. The receiving CPU must be a unit number n in the range 1-15 decimal.

If neither a user program nor an operating system is specified in the command line via local switch /S, the default operating system, SYS.SV/SYS.OL, is transmitted. The order of arguments in the command line is fixed; it cannot be varied.

Upon completion of this command, if an operating system was specified for transmission, the MCA bootstrap will be sent, followed by the system save and overlay files. If full initialization was specified via the /F switch, the CLI save and overlay files will also be transmitted as will all other specified files (filename ...). After all files have been transmitted, the system requests time and date information at the receiving operator's console (as occurs in disk bootstrapping; see Appendix E of the RDOS User's Manual, 093-000075). After this, the CLI gains control at the receiving CPU.

If instead of an operating system, a user program was transmitted, this program must conform to the conventions given for programs executable by HIPBOOT in Appendix E of the RDOS User's Manual, 093-000075. Such a user program will receive control upon being completely transmitted.

Filename arguments cannot be preceded by directory specifiers.

Name: MCABOOT (Continued)

Switches:

Global: /F - perform a full initialization (partial initialization with overlays is performed by default).

Local: /S - specify a user program or system save and overlay file other than the default system (SYS.SV/SYS.OL). The .SV extension need not be specified.

Asterisk: Not permitted.

Examples: MCABOOT/F MCAT:2 FORT.SV FIV.SV FORT.LB)

This command causes the default system CLI.SV and CLI.OL to be transmitted to unit 2 in the first MCA system, with full initialization. The transmitting CPU, also attached to the first MCA system, sends the FORTRAN IV compiler (FORT.SV and FIV.SV) and the FORTRAN IV runtime library. Since full initialization occurs, unit two also receives the CLI save and overlay files.

MCABOOT MCAT1:3 32K.SV/S)

This command causes an operating system, 32K.SV, and its associated overlay file, 32K.OL, to be transmitted to unit three, for a partial initialization with overlays. Both the transmitter and the receiver are attached to the second MCA system. The CLI save and overlay files are not sent since they were not specified in the command line and they are transmitted automatically only during full initialization.

HIGHER LEVEL PROGRAM COMMAND

This section includes the CLI command to return to the next high program level.

POP - To return to the next higher level program in this program environment.

Name: POP (Return to the next higher level program in this program environment.)

Format: POP)

Purpose: To return to the next higher level program in this program environment. Oftentimes a user program may wish to suspend its operation temporarily by pushing the CLI into operation at the next lower program level (via a .EXEC system call). The POP command provides a means to return from the lower level CLI to resume the user program's operation.

An attempt to issue this command from a CLI residing at program level zero will be rejected with a CLI runtime error.

Switches: None.

Asterisk: Not permitted.

Example: RESTORE.SV, a user program running at level 1, wishes to suspend its operation temporarily so that it can enlist the aid of the CLI to perform some routine maintenance function. However, upon the completion of this maintenance function, RESTORE wishes to resume its own operation. Thus a return (via CTRL A or any other means) to the level zero CLI would be inadequate, since it would not be possible to resume the operation of RESTORE at the point where it was suspended. To enlist the support of the CLI temporarily, RESTORE issues system call .EXEC, causing itself to be swapped to disk and the CLI to be invoked at level 2.

After using the CLI at level 2 to perform whatever functions were needed, the console operator merely issues the command

POP)

RESTORE will be restored in main memory and its execution will resume at the point where it was interrupted.

APPENDIX A

ERROR MESSAGES

ERROR	MEANING
ADDRESS ERROR	Attempt to reference an address outside user address space (mapped systems only).
ATTEMPT TO READ INTO SYSTEM SPACE	Attempted access of unmapped system area.
ATTEMPT TO WRITE AN EXISTING FILE	Attempt to write an existing file.
ATTEMPT TO RELEASE AN OPEN DEVICE	Attempt made to release an open device.
BRACKET ERROR	Nested or unmatched brackets.
CHECKSUM ERROR	Checksum error detected during input.
CHANNEL ALREADY IN USE	Attempt to open a channel which is already in use.
COMMON SIZE ERROR	The communications area specified for interprogram communications is too small.
COMMON USAGE ERROR	No communications area is defined in the other program (in a foreground/background environment).
CANNOT CHECKPOINT CURRENT BG	The current background is not checkpointable yet an attempt was made to checkpoint it. Conditions rendering a background non-checkpointable include the following: outstanding QTY I/O, user device interrupt service, outstanding read/write operator message, etc. Also, if one background program is currently checkpointed, the current background cannot be checkpointed.
CHANNEL CLOSED BY ANOTHER TASK	Two tasks share a common I/O channel. One task closes the channel before the other task was able to complete its I/O.
COMMAND LINE TO LONG	Command line exceeds limit.
DIRECT I/O ACCESS ONLY	Attempt to perform sequential I/O on a file with the I attribute.
DEVICE NOT IN SYSTEM	Attempt to reference an uninitialized directory or device.
DEVICE ALREADY IN SYSTEM	Illegal (and unnecessary) attempt to re-initialize a directory device. Alternatively, an attempt to identify a system device as a user device.

ERROR MESSAGES (Continued)

ERROR	MEANING
DIRECTORY SIZE INSUFFICIENT	An attempted CPART specified too few disk blocks.
DIRECTORY DEPTH EXCEEDED	Attempt to create a tertiary partition or a secondary subdirectory.
DIRECTORY IN USE	Attempt to assign a logical name to a directory or device which has already been initialized. Alternatively, an attempt to release a directory which has files in it that have not been closed.
DIRECTORY NOT INITIALIZED	Attempt to reference an uninitialized directory or device.
DIRECTORY SHARED	A released directory is in use by the other program (this is merely a warning).
DEVICE TIMEOUT	10 second disk timeout occurred in other than the master device.
DEVICE PREVIOUSLY OPENED	Attempt to open a previously opened device.
END OF FILE	End of file detected.
ERROR IN USER TASK QUEUE TABLE	Bad table input to .QTSK.
FILE READ PROTECTED	Attempt to read a read-protected file.
FILE WRITE PROTECTED	Attempt to modify a write-protected file.
FILE ALREADY EXISTS	Attempt to create a file with a name which is already in use.
FILE DOES NOT EXIST	Attempt to reference a non-existent file.
FILE ATTRIBUTE PROTECTED	Attempt to change the attributes of an attribute-protected file.
FILE NOT OPEN	Attempt to access an unopened file.
FILE SPACE EXHAUSTED	Out of user disk space or mag tape EOT reached before end of write.
FATAL SYSTEM UTILITY ERROR	An unrecoverable error was detected within a utility.
FILE DATA ERROR	File read error.

ERROR MESSAGES (Continued)

ERROR	MEANING
FILES MUST EXIST IN SAME DIRECTORY	Improper use of RENAME command.
FILE IN USE	Attempt to modify a file which is in use.
FOREGROUND ALREADY RUNNING	Attempt to execute a foreground program (EXFG) when one is already running.
FILE POSITION ERROR	Access attempted to save illegal position of a file.
ILLEGAL ARGUMENT	Illegal character in an argument.
ILLEGAL NUMERIC ARGUMENT	Non-numeric character in a numeric argument.
ILLEGAL ATTRIBUTE	Undefined attribute specified.
ILLEGAL BLOCK TYPE	Attempted LOAD of a file which is not in DUMP format.
ILLEGAL CHANNEL NUMBER	Channel number exceeding 77 octal was input to a system command.
ILLEGAL FILE NAME	Undefined character in file name string.
ILLEGAL SYSTEM COMMAND	Attempt to reference an uninitialized directory or device or to release an uninitializable device.
ILLEGAL COMMAND FOR DEVICE	Attempt to perform illegal I/O (e.g., direct I/O on disk data).
INSUFFICIENT MEMORY TO EXECUTE PROGRAM	Attempt to allocate more memory than is available.
ILLEGAL OVERLAY NUMBER	The specified overlay does not exist.
INVALID TIME CHANGE	Attempt to set illegal time or date.
INSUFFICIENT CONTIGUOUS BLOCKS	Insufficient number of free contiguous disk blocks to create the desired file.
ILLEGAL DIRECTORY NAME	Illegal character in link resolution name string was given in a LINK command.
INSUFFICIENT ROOM IN DATA CHANNEL MAP	Improper data channel size specified in .IDEF system command.

ERROR MESSAGES (Continued)

ERROR	MEANING
ILLEGAL PARTITION VALUE	One of the following conditions was detected in an SMEM command. The background and foreground allocation arguments do not equal the total user address space which is available; the new memory settings would not leave enough room for the background CLI (five 1024 word blocks are needed); the SMEM command was not issued from the background CLI while no foreground program was running.
LINK NOT ALLOWED	Attempt to link to a file which has the "no link resolution" attribute set.
LINK DEPTH EXCEEDED	Attempt to reference a resolution file via more than 10 levels of links.
LINE TOO LONG	Line limit exceeded on read or write line I/O.
MAP.DR ERROR	File system MAP.DR inconsistency detected in a directory device.
MCA REQUEST OUTSTANDING	No MCA receive request has been issued to correspond with an outstanding transmit request. Alternatively, an attempt has been made to post an MCA transmit or receive request on a channel which has an outstanding incomplete MCA transmission or reception.
NO MORE DCBS	Attempt to initialize too many devices and/or directories. SYSGEN a new system and specify a greater number of subdirectories/subpartition to be accessible at one time (question 18). Alternatively, release one or more currently initialized devices or directories.
NO DIRECT I/O	Direct I/O allowed only on magnetic or cassette tape files.
NO DEBUG ADDRESS	The debugger was not loaded with this save file and the DEB command was issued.
NO STARTING ADDRESS	Attempt to execute a non-save file.
NO ROOM FOR UFTS	Insufficient number of channels specified at SYSGEN time.
NOT A LINK ENTRY	Attempt to delete an entry which lacks the link characteristic. Attempting to DELETE a link causes the resolution file to be deleted.
NOT A SAVED FILE	File requires either the save attribute or the random characteristic.

ERROR MESSAGES (Continued)

ERROR	MEANING
NO SUCH DIRECTORY	Directory specifier unknown.
NO FILES MATCH SPECIFIER	No match found for any argument containing the * or - conventions.
NO SOURCE FILE SPECIFIED	CLI command requires a source file.
NOT A COMMAND	Fatal CLI error due to modification of CLI.SV or CLI.OL.
NOT ENOUGH ARGUMENTS	More arguments are required by this command.
NO MCA RECEIVE REQUEST OUTSTANDING	No outstanding receive request by an MCA device.
OUT OF TCB'S	Task Control Blocks all used.
PHASE ERROR	Attempt to reset location counter back over current value (only during MKSAVE: setting the counter ahead is permitted).
PERMANENT FILE	Attempt to delete or rename a permanent file; attempt to LOAD or DUMP a permanent file without using the /A global switch.
POP ERROR	Attempt to return from level zero.
PARITY ERROR	Parity error on read line or read sequential of mag tape.
PUSH DEPTH EXCEEDED	Attempt to push too many levels (limit of 4).
QTY ERROR	Simultaneous read or write attempt on same QTY line.
REPEAT ERROR	One, three or more parentheses in a single command line.
SPOOL FILES ACTIVE	System tried to relocate while spooling active.
SYSTEM DEADLOCK	System run out of resources i.e., buffers.
SIGNAL TO BUSY ADDRESS	Multiple transmissions to same address with no receive.

ERROR MESSAGES (Continued)

ERROR	MEANING
STACK OVERFLOW	Fatal CLI runtime error, generally caused by one of the following: CLI.OL has been modified; the creation dates of CLI.SV and CLI.OL do not match, command string to long. This latter condition might be caused by a user's attempting to replace a CLI while it was running, an impossible task since CLI.OL would be open and could not then be replaced.
SYSTEM STACK OVERFLOW	System stack capacity exceeded.
SQUASH FILE ERROR	Attempt to SQUASH a file that has already been squashed.
SYS.DR ERROR	File system SYS.DR inconsistency detected.
TASK ID ERROR	Task ID violates syntax requirement.
TRANSMISSION TERMINATED BY RECEIVER	The MCA transmission was terminated prematurely because its length exceeded that requested by the receiver.
TASK NOT FOUND FOR ABORT	Attempt to abort a task that does not exist.
TOO MANY ACTIVE DEVICES	Too many devices active at the same time.
TOO MANY ARGUMENTS	Too many arguments input to a command where the number or position of arguments is significant.
UNIT IMPROPERLY SELECTED	Magnetic or cassette tape controller not turned on, tape unit not ON LINE. Disk unit adapter not turned on.
UNMATCHED @	Command line has an odd number of @ signs.
YOU CAN'T DO THAT	Attempt to end console logging without using password. Attempt to run an ECLIPSE program on a NOVA. Alternatively, this message is given to the CLI (e.g., non-upper case ASCII characters.)

APPENDIX B

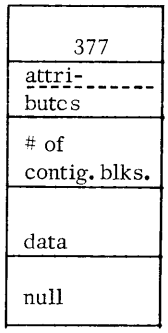
DUMP FILE FORMAT

Dump files are formatted into the following types of blocks:

1)	Name blocks	[377]
2)	Data blocks	[376]
3)	Error blocks	[375]
4)	End blocks	[374]
5)	Time blocks	[373]
6)	Link data blocks	[372]
7)	Link Access Attribute blocks	[371]
8)	End of Segment blocks	[370]

These formats are given in the following illustrations:

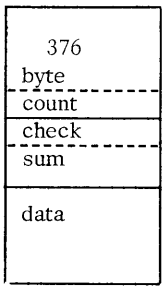
1. NAME BLOCK



	size	contents
377	1 byte	block type identifier
attri-butes	2 bytes	attributes
# of contig. blks.	2 bytes	no. of contiguous blocks only if contiguous characteristic set
data	VARIABLE	filename
null		

NOTE: filename is written by means of system call .WRS, so if reading from paper tape, you must use system call .RDS to avoid parity problems.

2. DATA BLOCK



	size	contents
376	1 byte	block type identifier
byte count	2 bytes	byte count
check sum	2 bytes	checksum (word count mod 2 + total contents of all full words.)
data	n bytes	data (file contents)

no terminator

NOTE: Any odd byte is written in "data" and is added in "byte count" but it is not summed. There is no end-around carry.

DUMP FILE FORMAT (Continued)

3. ERROR BLOCK

	size	contents
375	1 byte	block type identifier

NOTE: This block causes the file currently being loaded to be read through to the end but not to be written out.

4. END BLOCK

	size	contents
374	1 byte	block type identifier

NOTE: Only one of these appears in each dump file, with the exception of subdirectories, which are also terminated by an end block. There is no end block for each file dumped to one dump file.

5. TIME BLOCK (follows name block except on links)

	size	contents
373	1 byte	block type identifier
last access	2 bytes	Julian day last accessed (opened) since Jan. 1, 1968 (day 1)
day created	2 bytes	Julian day created
time created	2 bytes	left byte = hour, right byte = minute created

no terminator

6. LINK DATA BLOCK (follows name block on link)

	size	contents
372	1 byte	block type identifier
	variable	alternate directory name (if any) terminated by null.
null		
	variable	link alias name (if any) terminated by null
null		(minimum of 2 nulls if link has neither alternate directory nor alias)

NOTE: These 2 names are written via system call .WRL, so if reading from paper tape, use .RDL to avoid parity problems.

7. LINK ACCESS ATTRIBUTES BLOCK

	size	contents
371	1 byte	block type identifier
attributes	2 bytes	link access attributes

DUMP FILE FORMAT (Continued)

8. END OF SEGMENT BLOCK

	size	contents
370	1 byte	block type identifier
-1	2 bytes	-1
number	1 byte	segment number
data	VARIABLE	filename
null		

*****

APPENDIX C

CLI INTERPRETATION OF KEYBOARD COMMANDS

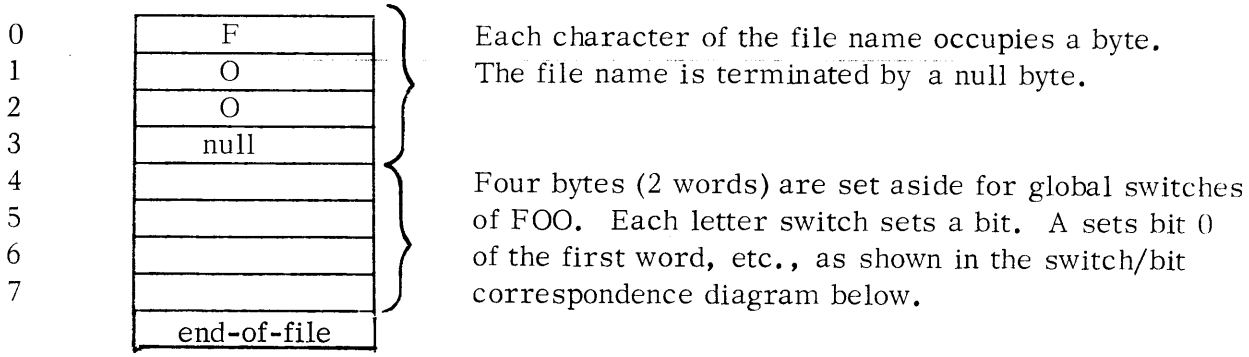
The action taken by the CLI upon reading a command line is sufficiently flexible so that users can, if they wish, design programs to perform system command functions.

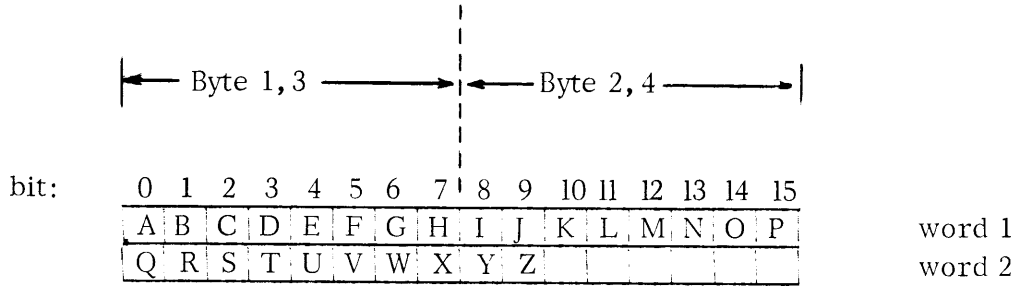
When either the background or foreground CLI reads a command line and does not recognize the first file name, the CLI always builds a command file before the save file of that name is loaded. The command file reflects an edited version of the command line. The name of the command file is COM.CM if it is built by the background CLI. The foreground CLI builds a command file identical in structure to COM.CM, but the foreground command file is named FCOM.CM.

For example, suppose the user issues the command line:

FOO)

The CLI does not recognize FOO as a known command word. It therefore builds a command file with the byte organization shown below:





Bit 15 of the second word in the global switches field is always set for files running under BATCH. This serves to indicate that it is running under BATCH instead of the CLI.

Note that the CLI does not attempt any interpretation of switches in building the command file. The CLI simply sets the appropriate bit.

Additional file name arguments and local switches are handled in the same way when the CLI builds the command file. Suppose the user types the command:

```
FOO/B AA ZZ/X MUMB )
```

The CLI would then build the following command file:

Byte	Contents	
0	F	Command file name FOO, terminated by null byte.
1	O	
2	O	
3	null	
4	1	Global switches of FOO. Bit 1 (switch B) set ON.
5		
6		
7		
8	A	Argument AA, terminated by null byte.
9	A	
10	null	
11		Four bytes set aside for local switches of AA. None set.
12		
13		
14		
15	Z	Argument ZZ, terminated by null byte.
16	Z	
17	null	
18		Local switches of ZZ. Bit 23 (switch X) set ON.
19		
20	1	
21		
22	M	Argument MUMB, terminated by null byte.
23	U	
24	M	
25	B	
26	null	
27		Local switches of MUMB. None set.
28		
29		
30		

Since the CLI does not interpret switches, the user can set up program interpretation of such switches. This gives the user an added means of passing information to a program to be executed, since he can use switches as well as arguments.

A read line from a disk file will terminate on a null (as well as carriage return and form feed). This is quite useful in reading COM.CM and FCOM.CM arguments. The following example illustrates how a background user could read the first argument of the command file as well as its global switches.

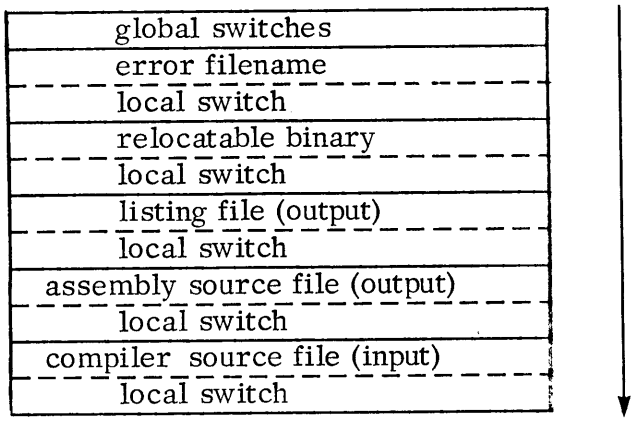
```
        LDA     0,CFILE     ;COM.CM POINTER
        .SYSTM
        .OPEN   3           ;OPEN ON CHANNEL 3
        JSR     EROR        ; ??
        LDA     0, ARG1     ;FILE NAME AREA POINTER
        .SYSTM              ;READ IT (THE NULL
        .RDL    3           ;TERMINATOR IS ALSO
        JSR     EROR        ;TRANSFERRED)
        LDA     0,GLOB      ;POINTER FOR GLOBAL
        LDA     1,C4        ;SWITCHES
        .SYSTM              ;READ FOUR BYTES
        .RDS    3
        JSR     EROR
        .
        .
        .
C4:     4
GLOB:   2*.GLOB
ARG1:   2*.ARG1
CFILE:  2*.CFIL

.GLOB   .BLK    2
.ARG1   .BLK    10
.CFIL   .TXT    *COM.CM*
```

When square brackets are detected in a command line, they are passed as a file name with bit 10 in the second word of switch information set (this bit follows the Z switch position). Commas within square brackets are treated in the same manner (bit 10 set).

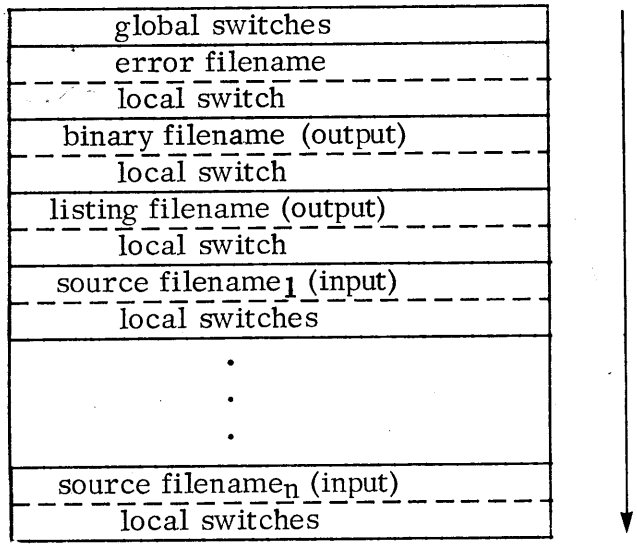
Utility Program Command File Structures

Following is a series of illustrations depicting the command file structures for all system utilities available via the CLI.

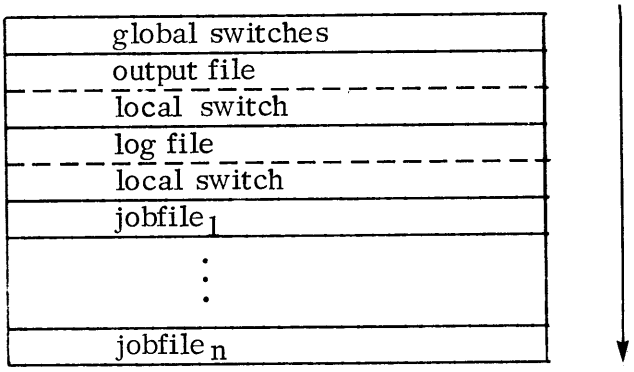
ALGOL



ASM

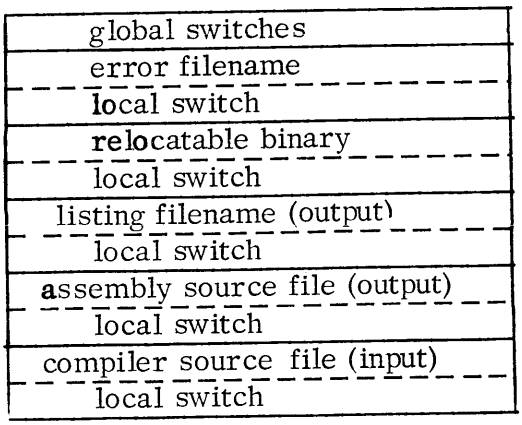


BATCH

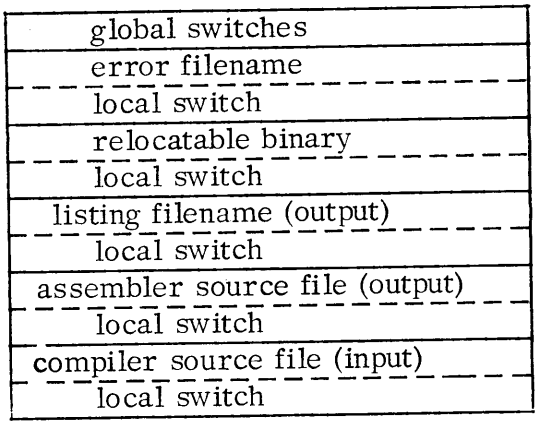


Utility Program Command File Structures (Continued)

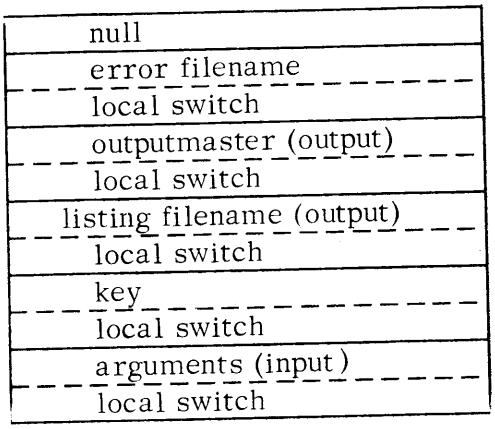
FORT



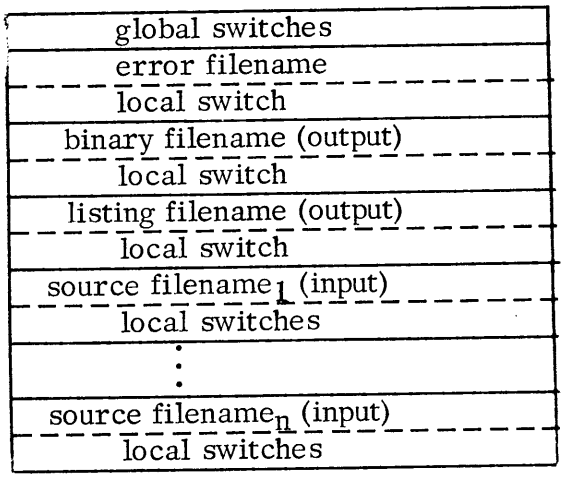
FORTRAN



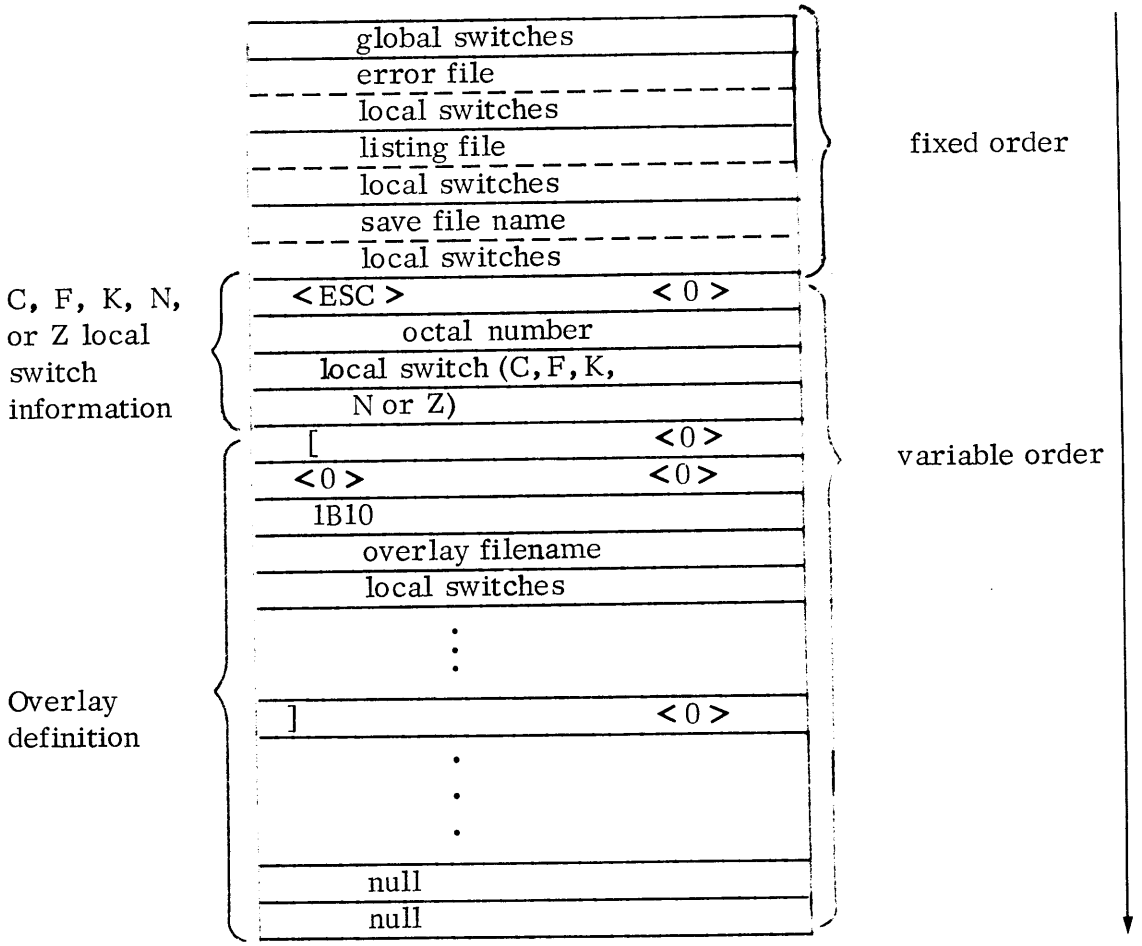
LFE



MAC


global switches
error filename
local switch
binary filename (output)
local switch
listing filename (output)
local switch
source filename$_1$ (input)
local switches
source filename$_n$ (input)
local switches

RLDR

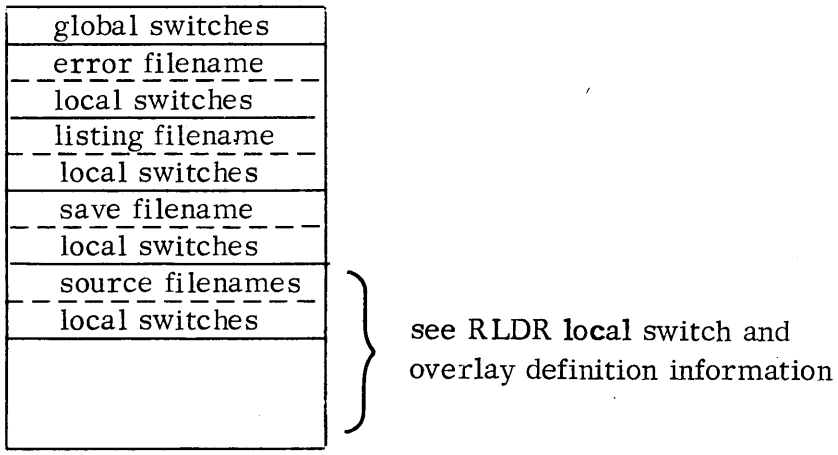

global switches
error file
local switches
listing file
local switches
save file name
local switches
fixed order
C, F, K, N, or Z local switch information
<ESC> <0>
octal number
local switch (C, F, K, N or Z)
[<0>
<0> <0>
1B10
overlay filename
local switches
Overlay definition
] <0>
null
null
variable order

Given the overlay definition [A, B C], the overlay definition field of the command file would look like the following:

```
A<0>
<0> <0>
<0> <0>
, <0>
<0> <0>
1B10
B<0>
<0> <0>
<0 > <0>
C<0>
<0> <0>
<0> <0>
```

Local switch information fields (C, F, K, N, or Z) and overlay definition fields may occur in any order following the fixed portion of the command file.

CLG



Source filenames are listed in the order that they appear in the command line; source filenames receive the extension .FR by default. The following local switch operations occur:

- /A - source file argument is given the extension .SR .
- /C - channel argument is converted to octal.
- /K - task argument is converted to octal.
- /O - source file argument is given the extension .RB .

DataGeneral

SOFTWARE DOCUMENTATION
REMARKS FORM

Document Title	Document No.	Tape No.

SPECIFIC COMMENTS: List specific comments. Reference page numbers when applicable. Label each comment as an addition, deletion, change or error if applicable.

GENERAL COMMENTS: Also, suggestions for improvement of the Publication.

FROM:

Name	Title	Date

Company Name

Address (No. & Street)	City	State	Zip Code

FOLD DOWN FIRST FOLD DOWN

FIRST
CLASS
PERMIT
No. 26
Southboro
Mass. 01772

BUSINESS REPLY MAIL

No Postage Necessary If Mailed In The United States

Postage will be paid by:

Data General Corporation

Southboro, Massachusetts 01772

ATTENTION: Software Documentation

FOLD UP SECOND FOLD UP

STAPLE