NAME nm -- get name list

SYNOPSIS nm [name]

DESCRIPTION nm prints the symbol table from the output file of an assembler or loader run. Only relocatable, global, and undefined symbols-- not absolute-- are given. Each defined symbol is preceded by its value; each undefined symbol by blanks. Global symbols have their first character underlined. The output is sorted alphabetically.

If no file is given, the symbols in a.out are listed.

FILES a.out

SEE ALSO as, ld

DIAGNOSTICS` "?"

BUGS

OWNER dmr, ken

NAME od -- octal dump

SYNOPSIS od name [origin]

DESCRIPTION od dumps a file in octal, eight words per line with the origin of the line on the left. If an octal origin is given it is truncated to 0 mod 16 and dumping starts from there, otherwise from 0. Printing continues until halted by sending an interrupt signal.

FILES

SEE ALSO db

DIAGNOSTICS

BUGS Dumping does not cease at the end of the file; instead the file appears to be padded with garbage to a length of 511 mod 512 bytes.

OWNER ken, dxnr

NAME pr -- print file

SYNOPSIS pr [—lcm] name1 ...

DESCRIPTION produces a printed listing of one or more files. The output is separated into pages headed by the name of the file, a date, and the page number.

The optional flag —l causes each page to contain 78 lines instead of the standard 66 to accommodate legal size paper.

The optional flags —c (current date) and —m (modified date) specify which date will head all subsequent files. —m is default.

FILES /dev/ttyn to suspend messages.

SEE ALSO cat, cp, mesg

DIAGNOSTICS -- (files not found are ignored)

BUGS none

OWNER ken, dmr

NAME rew -- rewind tape

SYNOPSIS rew [digit]

DESCRIPTION rew rewinds DECtape drives. The digit is the logical tape number, and should range from 0 to 7. A missing digit indicates drive 0.

FILES /dev/tap0, ..., /dev/tap7

SEE ALSO

DIAGNOSTICS "?" if there is no tape mounted on the indicated drive or if the file cannot be opened.

BUGS

OWNER ken, dmr

NAME rkd -- dump RK disk to tape

SYNOPSIS /etc/rkd

DESCRIPTION _rkd_ copies an RK03/RK05 disk pack onto nine DECtapes.

Physical I/O is done and interrupts are disabled, so time-sharing is suspended during operation of the command.

The sequence of tape drives is: 0, 1, 0, 1, ...

rkd exits if 0 appears in the console switches.

FILES --

SEE ALSO rkl

DIAGNOSTICS none; errors are retried forever

BUGS

OWNER ken

NAME rkf -- format RK03 disk pack

SYNOPSIS rkf

DESCRIPTION rkf formats a virgin disk pack. Because it destroys all information on that pack, and because it is not interlocked against file system activity on the pack, the rkf program is not maintained in executable form. Instead the source must be located and assembled.

FILES none (uses physical I/O on drive 0).

SEE ALSO

DIAGNOSTICS "error" is printed and a core image is produced if a write error occurs. A copy of the RK status register is in register 5.

BUGS As mentioned, is not interlocked against system I/O; if I/O is already occurring, it will be badly disrupted. In any event, all information on the pack is destroyed.

OWNER ken, dmr

NAME rkl -- reload RK disk from tape

SYNOPSIS /etc/rkl

DESCRIPTION _rkl_ loads an RK03/RK05 disk pack from nine DECtapes.

The program uses physical I/O with interrupts disabled; therefore time—sharing is suspended.

Only the super—user may invoke this command.

The sequence of drives is: 0, 1, 0, 1, ... _rkl_ will cease if 0 appears in the console switches.

FILES

SEE ALSO rkd

DIAGNOSTICS none; errors are retried forever

BUGS --

OWNER ken

NAME rm -- remove (unlink) files

SYNOPSIS' rm name1 ...

DESCRIPTION rm removes the entries for one or more files from a directory. If an entry was the last link to the file, the file is destroyed. Removal of a file requires write permission in its directory, but neither read nor write permission on the file itself.

Directories cannot be removed by rm; cf. rmdir.

FILES none.

SEE ALSO rmdir, for removing directories.

DIAGNOSTICS If the file cannot be removed or does not exist, the name of the file followed by a question mark is typed.

BUGS rm probably should ask whether a read—only file is really to be removed.

OWNER ken, dmr

NAME rmdir -- remove directory

SYNOPSIS rmdir dir1 ...

DESCRIPTION rmdir removes (deletes) directories. The directory must empty (except for the standard entries "." and "..", which rmdir itself removes). Write permission is required in the directory in which the directory appears.

FILES none

SEE ALSO

DIAGNOSTICS "dir?" is printed if directory dir cannot be found, is not a directory, or is not removable.

"dir -- directory not empty" is printed if dir has entries other than "." or "..".

BUGS

OWNER ken, dmr

NAME roff -- format text

SYNOPSIS roff [+number] [—number] name1 ...

DESCRIPTION roff formats text according to control lines embedded in the text. The optional argument "+number" causes printing to begin at the first page with the appropriate number; —number causes printing to cease at the first page with a higher number.

roff is fully described in a separate publication [reference].

FILES /etc/suftab contains a list of suffixes used to guide hyphenation. /tmp/rtma, rtmb, ... temporary /dev/ttyn to suspend messages.

SEE ALSO [reference], mesg

DIAGNOSTICS: none -- files not found are ignored

BUGS roff does not check for various kinds of buffer overflow. If a fault occurs, check the input in the region where the error occurred.

OWNER jfo, dmr, ken

NAME sdate -- set date and time

SYNOPSIS sdate mmddhhmm

DESCRIPTION sdate adjusts the system's idea of the date and time. mm is the month number; dd is the day number in the month; hh is the hour number (24-hour system); mm is the minute number. For example,

```
sdate 10080045
```

sets the date to Oct. 8, 12:45 AM.

FILES none

SEE ALSO date

DIAGNOSTICS "?" if the date is syntactically incorrect.

BUGS none

OWNER ken, dmr

NAME sh -- shell (command interpreter)

SYNOPSIS sh [name [arg1 ... [arg9]]]

DESCRIPTION sh is the standard command interpreter. It is the program which reads and arranges the execution of the command lines typed by most users. It may itself be called as a command to interpret files of command lines. Before discussing the arguments to the shell used as a command, the structure of command lines themselves will be given.

Command lines are sequences of commands separated by command delimiters. Each command is a sequence of non—blank command arguments separated by blanks. The first argument specifies the name of a command to be executed. Except for certain types of special arguments discussed below, the arguments other than the command name are simply passed to the invoked command.

If the first argument represents the path name of an executable file, it is invoked; otherwise the string "/bin/" is prepended to the argument. (In this way the standard commands, which reside in "/bin," are found.) If this search too fails a diagnostic is printed.

The remaining non—special arguments are simply passed to the command without further interpretation by the shell.

There are three command delimiters: the new line, ";" , and "&". The semicolon ";" specifies sequential execution of the commands so separated; that is,

```
coma; comb
```

causes the execution first of command coma, then of comb. The ampersand "&" causes simultaneous execution:

```
coma & comb
```

causes coma to be called, followed immediately by comb without waiting for coma to finish. Thus coma and comb execute simultaneously. As a special case,

```
coma &
```

causes coma to be executed and the shell immediately to request another command without waiting for coma.

Two characters cause the immediately following string to be interpreted as a special argument to the shell itself, not passed to the command. An argument of the form <arg causes the file arg to be used as the standard input file of the given command; an argument of the form ">arg" causes file "arg" to be used as the standard output file for the given command.

If any argument contains either of the characters "?" or "*" , it is treated specially as follows. The current directory is searched for files which match the given argument. The character "*" in an argument matches any string of characters in a file name (including the null string); ? matches any single character in a file name. Other argument characters match only the same character in the file name. For example, "*" matches all file names; "?" matches all one—character file names; "ab*.s" matches all file names beginning with "ab" and ending with ".s".

If the argument with "*" or "?" also contains a "/", a slightly different procedure is used: instead of the current directory, the directory used is the one obtained by taking the argument up to the last "/" before a "*" or "?". The matching process matches the remainder of the argument after this "/" against the files in the derived directory. For example:, "/usr/dmr/a*.s" matches all files in directory /usr/dmr which begin with "a" and end with ".s"

In any event, a list of names is obtained which match the argument. This list is sorted into alphabetical order, and the resulting sequence of arguments replaces the single argument containing the "*" or "?". The same process is carried out for each argument with a * or ?" (the resulting lists are not merged) and finally the command is called with the resulting list of arguments..

For example: directory /usr/dmr contains the files al • s, a2.s, ..., a9.s. From any directory, the command

as /usr/dmr/a?.s

calls as with arguments /usr/dmr/al.s, /usr/dmr/a2.s, ... /usr/dmr/a9.s in that order.

The character "\" causes the immediately following character to lose any special meaning it may have to the shell; in this way < , , and other characters meaningful to the shell may be passed as part of arguments. A special case of this feature allows the continuation of commands onto more than one line: a new–line preceded by "\" is translated into a blank.

Sequences of characters enclosed in double (") or single (') quotes are also taken literally.

When the shell is invoked as a command, it has additional string processing capabilities. Recall that the form in which the whell is invoked is

```
sh [ name [ arg1 ... [ arg9 ] ] ]
```

The name is the name of a file which will be read and interpreted. If not given, this subinstance of the shell will continue to read the standard input file.

In the file, character sequences of the form "$n", where <u>n</u> is a digit 0, ..., 9, are replaced by the nth argument to the invocation of the shell (arg). $0" is replaced by <u>name</u>.

An end–of–file in the shell's input causes it to exit. A side effect of this fact means that the way to log out from UNIX is to type an end of file.

FILES /etc/glob

SEE ALSO [reference], which gives the theory of operation of the shell.

DIAGNOSTICS "?", in case of any difficulty. The most common problem is inability to find the given command. Others: input file ("<") cannot be found; no more processes can be created (this will alleviate itself with the passage of time). Note that no diagnostic is given for inability to create an output (">") file; the standard output file has already been closed when the condition is discovered and there is no place to write the diagnostic.

If a "*" or "?" is used, the "glob" routine is invoked; it types "No command" if it cannot find the given command, and "No match" if there were no files which matched an argument with "?" or "*".

BUGS Better diagnostics should be provided. If a "*" or "?" is used, the command must be in /<u>bin</u> (Not, for example, in the user's directory.) This is actually a <u>glob</u> bug.

OWNER dmr, ken

NAME stat -- get file status

SYNOPSIS stat name1

DESCRIPTION stat gives several kinds of information about one or more files:

```
i—number
access mode
number of links
owner
size in bytes
date and time of last modification
name (useful when several files are named)
```

All information is self—explanatory except the mode. The mode is a six—character string whose characters mean the following:

```
1  s: file is small (smaller than 4096 bytes)
   l: file is large

2  d: file is a directory
   x: file is executable
   u: set user ID on execution
   —: none of the above

3  r: owner can read
   —: owner cannot read

4 w: owner can write
   —: owner cannot write

5  r: non—owner can read
   —: non—owner cannot read

6  w: non—owner can write
   —: non—owner cannot write
```

The owner is almost always given in symbolic form; however if he cannot be found in "/etc/uids" a number is given.

If the number of arguments to stat is not exactly 1 a header is generated identifying the fields of the status information.

FILES /etc/uids

SEE ALSO ls with the —l option gives the same information as stat.

DIAGNOSTICS "name?" for any error.

BUGS none

NAME strip -- remove symbols and relocation bits

SYNOPSIS strip name1

DESCRIPTION strip removes the symbol table and relocation bits ordinarily attached to the output of the assembler and loader. This is useful to save space after a program has been debugged.

The effect of strip is the same as use of the —s option of ld.

FILES /tmp/stma, stmb ... temporary file

SEE ALSO ld, as

DIAGNOSTICS Diagnostics are given for: non—existent argument; inability to create temporary file; improper format (not an object file); inability to re—read temporary file.

BUGS

OWNER dmr

NAME su -- become privileged user

SYNOPSIS su password

DESCRIPTION su allows one to become the super—user, who has all sorts of marvelous powers. In order for su to do its magic, the user must pass as an argument a password. If the password is correct, su will execute the shell with the UID set to that of the super—user. To restore normal UID privileges, type an end—of—file to the super—user shell

FILES

SEE ALSO shell

DIAGNOSTICS "Sorry" if password is wrong

BUGS

OWNER dmr, ken

NAME sum -- sum file

SYNOPSIS sum name

DESCRIPTION sum sums the contents of a file. In practice, it is most often used to verify that all of a DECtape can be read without error.

FILES none

SEE ALSO

DIAGNOSTICS "?" if the file cannot be read at all or if an error is discovered during the read.

BUGS none

OWNER ken

NAME tap -- manipulate DECtape

SYNOPSIS tap [key] [name ...]

DESCRIPTION <u>tap</u> saves and restores selected portions of the file system hierarchy on DECtape. Its actions are controlled by the key argument. The key is a string of characters containing at most one function letter and possibly one or more function modifiers. Other arguments to the command are file or directory names specifying which files are to be dumped, restored, or tabled.

The function portion of the key is specified by one of the following letters:

r The indicated files and directories, together with all subdirectories, are dumped onto the tape. If files with the same names already exists, they are replaced (hence the r). "Same" is determined by string comparison, so "./abc" can never be the same as "/usr/dmr/abc even if "/usr/dmr" is the current directory. If no file argument is given, "/" is the default.

u updates the tape. <u>u</u> is the same as r, but a file is replaced only if its modification date is later than the date stored on the tape; that is to say, if it has changed since it was dumped. <u>u</u> is the default command if none is given.

d deletes the named files and directories from the tape. At least one file argument must be given.

x extracts the named files from the tape to the file system. The owner, mode, and date—modified are restored to what they were when the file was dumped. If no file argument is given, the entire contents of the tape are extracted.

t lists the names of all files stored on the tape which are the same as or are hierarchically below the file arguments. If no file argument is given, the entire contents of the tape are tabled.

1 is the same as <u>t</u> except that an expanded listing is produced giving all the available information about the listed files.

The following characters may be used in addition to the letter which selects the function desired.

0, ..., 7 This modifier selects the drive on which the tape is mounted. "0" is the default.

v Normally tap does its work silently. The v (verbose) option causes it to type the name of each file it treats preceded by a letter to indicate what is happening.

r file is being replaced
a file is being added (not there before)
x file is being extracted
d file is being deleted

The v option can be used with r, u, d, and x only.

c means a fresh dump is being created; the tape directory will be zeroed before beginning. Usable only with r and u.

f causes new entries copied on tape to be 'fake' in that only the entries, not the data associated with the entries are updated. Such fake entries cannot be extracted. Usable only with r' and u.

w causes tap to pause before treating each file, type the indicative letter and the file name (as with v) await the user's response. Response "y" means "yes", so the file is treated. Null response means "no" , and the file does not take part in whatever is being done. Response "x" means exit ; the tap command terminates immediately. In the x function, files previously asked about have been extracted already. With r, u and d no change has been made to the tape.

m make (create) directories during an x if necessary.

i ignore tape errors. It is suggested that this option be used with caution to read damaged tapes.

FILES /dev/tap0 ... /dev/tap7

SEE ALSO rk

DIAGNOSTICS RK open error
RK read error
RK write error
Directory checksum
Directory overflow

RK overflow

Phase error (a file has changed after it was selected for dumping but before it was dumped)

BUGS All references to "RK" should read "tape." The m option does not work correctly in all cases. The i option is not yet implemented.

OWNER ken

NAME tm -- provide time information

SYNOPSIS _tm_ [command arg1 ....]

DESCRIPTION _tm_ is used to provide timing information. When used without an argument, output like the following is given:

```
tim                    77:43:20  29.2
ovh         13:59:42    1.2
dsk         12:06:30    4.1
idl        352:31:37   23.7
usr          3:32:15    0.1
der         5, 171   0,  0
```

The first column of numbers gives totals in the named categories since the last time the system was cold—booted; the second column gives the changes since the last time _tm_ was invoked. The _tim_ row is total real time (hours:minutes:seconds); unlike the other times, its origin is the creation date of tm's temporary file. _ovh_ is time spent executing in the system; dsk is time spent waiting for both kinds of disk I/O; idl is idle time; usr is user execution time; _der_ is RF disk error count (left number) and RK disk error count (right number).

_tm_ can be invoked with arguments which are assumed to constitute a command to be timed. In this case the output is as follows:

```
tim         2.2
ovh         0.3
dsk         1.8
idl         0.0
usr         0.0
```

The given times represent the number of seconds spent in each category during execution of the command.

FILES /tmp/ttmp, /dev/rf0 (for absolute times) contains the information used to calculate the differential times.

SEE ALSO format of file system (which tells where the times come from)

DIAGNOSTICS "?" if the command cannot be executed; "can't creat temp file" if trouble with /_tmp_; "cant read super—block" if times cannot be read from system.

BUGS (1) when invoked with a command argument, everything going on at the moment is counted, not just the command itself. (2) Two users doing _tm_ simultaneously interfere with each other's use of the temporary file.

OWNER ken, dmr

NAME tty -- get tty name

SYNOPSIS tty

DESCRIPTION tty gives the name of the user's typewriter in the form "ttyn" for n a_digit. The actual path name is then "/dev/ttyn".

FILES

SEE ALSO

DIAGNOSTICS "not a tty" if the standard input file is not a typewriter.

BUGS

OWNER dmr, ken

NAME type -- type on 2741

SYNOPSIS _type_ name1

DESCRIPTION _type_ produces output on an IBM 2741 terminal with a Correspondence type ball.

type uses typewriter tty5, which, because of the lack of access ports, is also used as a standard communication channel. Therefore, _who_ should be used to verify the absence of a user on tty5.

The method is as follows: type the _type_ command. It will wait until tty5 is dialled up. When the phone answers, depress the interrupt button after paper has been loaded, and the first file will be typed. spaces out to the end of a sheet of paper and waits until the interrupt button is depressed before beginning each new file.

FILES /dev/tty5

SEE ALSO who

DIAGNOSTICS

BUGS obviously some scheme is needed to prevent interference between normal users and The best thing would be to support 2741's as a standard terminal.

OWNER dmr

NAME umount -- dismount file system

SYNOPSIS <u>umount</u> special

DESCRIPTION announces to the system that the removable file system previously mounted on special file <u>special</u> is to be removed.

Only the super—user may issue this command.

FILES

SEE ALSO mount

DIAGNOSTICS

BUGS This command should be restricted to the super— user.

OWNER ken, dmr

NAME un -- undefined symbols

SYNOPSIS un [name]

DESCRIPTION un prints a list of undefined symbols from an assembly or loader run. If the file argument is not specified, a out is the default. Names are listed alphabetically except that non–global symbols come first. Undefined global symbols (unresolved external references) have their first character underlined.

FILES a.out

SEE ALSO as, ld

DIAGNOSTICS "?" if the file cannot be found.

BUGS

OWNER dmr, ken