

ACADEMIC OPERATING SYSTEM 4.3

ACADEMIC OPERATING SYSTEM



VOLUME I



ACADEMIC OPERATING SYSTEM 4.3

ACADEMIC OPERATING SYSTEM



VOLUME I

IBM Academic Operating System 4.3

PRPQ #5799-WZQ PRPQ #5799-PFF

Volume I

Academic Information Systems
International Business Machines Corporation
Palo Alto, CA

NOTICE

This product is distributed by way of license and is copyright protected. Unauthorized copying or reproduction by any means of any part of this product is expressly prohibited, except as provided by the license agreement.

WARNING: Any shipment of Academic Operating System 4.3 (hereinafter referred to as "IBM/4.3") for the IBM RT PC or IBM 6152 Academic System to a country outside the United States requires a U.S. Government license.

First Edition (December 1987)

Changes will be made periodically to the information herein; these changes will be incorporated in new editions of this publication.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available outside the United States.

International Business Machines Corporation provides this software and documentation "as is", without warranty of any kind, either express or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. No assurance of successful installation can be given. IBM may make improvements and/or changes in the product(s) and/or program(s) described in this manual at any time.

A reader's comment form has been provided at the back of this publication. If the form has been removed, address comments to Academic Information Systems; University Support, Dept. 6FR; IBM Corporation; P.O. Box 10500; Palo Alto, CA 94303.

IBM may use or distribute any of the information you supply in any way it deems appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1985, 1986, 1987

TRADEMARKS

The following trademarks appear in this manual:

UNIX is a registered trademark of AT&T Bell Laboratories

DEC and VAX are trademarks of Digital Equipment Corporation

AIX, RT, RT PC, and Personal System/2 are trademarks of International Business Machines Corporation

MetaWare, High C, and Professional Pascal are trademarks of MetaWare Incorporated

Ethernet is a trademark of Xerox Corporation

Documenter's Workbench is a registered trademark of AT&T Technologies

This page intentionally left blank.

Preface

This software and documentation is based in part on the 4.3 Berkeley Software Distribution under license from The Regents of the University of California. We gratefully acknowledge the following individuals and institutions for their role in its development: Computer Science Division, Department of Electrical Engineering and Computer Science, University of California, Berkeley, California; Individual Computing Systems, IBM Research, Yorktown Heights, New York; The IRIS Group, Brown University, Providence, Rhode Island; ITC, Carnegie-Mellon University, Pittsburgh, Pennsylvania.

IBM Academic Information Systems Palo Alto, CA July 1987

Preface to 4.3 Berkeley Software Distribution

This update to the 4.2 distribution of August 1983 provides substantially improved performance, reliability, and security, the addition of Xerox Network System (NS) to the set of networking domains, and partial support for the VAX 8600 and MICROVAXII.

We were greatly assisted by the DEC UNIX Engineering group who provided two full time employees, Miriam Amos and Kevin Dunlap, to work at Berkeley. They were responsible for developing and debugging the distributed domain based name server and integrating it into the mail system. Mt Xinu provided the bug list distribution service as well as donating their MICROVAXII port to 4.3BSD. Drivers for the MICROVAXII were done by Rick Macklem at the University of Guelph. Sam Leffler provided valuable assistance and advice with many projects. Keith Sklower coordinated with William Nesheim and J. Q. Johnson at Cornell, and Chris Torek and James O'Toole at the University of Maryland to do the Xerox Network Systems implementation. Robert Elz at the University of Melbourne contributed greatly to the performance work in the kernel. Donn Seeley and Jay Lepreau at the University of Utah relentlessly dealt with a miriad of details; Donn completed the unfinished performance work on Fortran 77 and fixed numerous C compiler bugs. Ralph Campbell handled innumerable questions and problem reports and had time left to write rdist. George Goble was invaluable in shaking out the bugs on his production systems long before we were confident enough to inflict it on our users. Bill Shannon at Sun Microsystems has been helpful in providing us with bug fixes and improvements. Tom Ferrin, in his capacity as Board Member of Usenix Association, handled the logistics of large-scale reproduction of the 4.2BSD and 4.3BSD manuals. Mark Seiden helped with the typesetting and indexing of the 4.3BSD manuals. Special mention goes to Bob Henry for keeping ucbvax running in spite of new and improved software and an ever increasing mail, news, and uucp load.

Numerous others contributed their time and energy in creating the user contributed software for the release. As always, we are grateful to the UNIX user community for encouragement and support.

Once again, the financial support of the Defense Advanced Research Projects Agency is gratefully acknowledged.

M. K. McKusick M. J. Karels J. M. Bloom

Preface to 4.2 Berkeley Software Distribution

This update to the 4.1 distribution of June 1981 provides support for the VAX 11/730, full networking and interprocess communication support, an entirely new file system, and many other new features. It is certainly the most ambitious release of software ever prepared here and represents many man-years of work. Bill Shannon (both at DEC and at Sun Microsystems) and Robert Elz of the University of Melbourne contributed greatly to this distribution through new device drivers and painful debugging episodes. Rob Gurwitz of BBN wrote the initial version of the code upon which the current networking support is based. Eric Allman of Britton-Lee donated countless hours to the mail system. Bill Croft (both at SRI and Sun Microsystems) aided in the debugging and development of the networking facilities. Dennis Ritchie of Bell Laboratories also contributed greatly to this distribution, providing valuable advise and guidance. Helge Skriverik worked on the device drivers which enabled the distribution to be delivered with a TU58 console cassette and RX01 console floppy disk, and rewrote major portions of the standalone i/o system to support formatting of non-DEC peripherals.

Numerous others contributed their time and energy in organizing the user software for release, while many groups of people on campus suffered patiently through the low spots of development. As always, we are grateful to the UNIX user community for encouragement and support.

Once again, the financial support of the Defense Advanced Research Projects Agency is gratefully acknowledged.

S. J. Leffler W. N. Joy M. K. McKusick

CONTENTS

SUMMARY OF AMENDMENTS

ABOUT THIS MANUAL

VOLUME I. MANUAL PAGES

Permuted Index of Revised/New Manual Pages

Section 1. Commands and Application Programs

Section 2. System Calls

Section 3. Subroutines

Section 4. Special Files

Section 5. File Formats and Conventions

Section 8. Maintenance Commands and Procedures

PERMUTED INDEX

IBM/4.3 Contents

This page intentionally left blank.

SUMMARY OF AMENDMENTS

This section summarizes the changes in each update and revision of IBM Academic Operating System 4.3. The changes are arranged by date, with the most recent given first. Technical Highlights are followed by Documentation Changes.

July 1987

1. TECHNICAL HIGHLIGHTS

4.3 Berkeley Software Distribution (4.3BSD) Upgrade

The 4.3/RT operating system is functionally equivalent to 4.3BSD while keeping all IBM extensions available in IBM Academic Information Systems 4.2 for the IBM RT PC.

General changes:

- This system has full 4.3BSD capabilities, with the exceptions noted in Appendix A. The articles "Bug Fixes and Changes in 4.3BSD" and "Changes to the Kernel in 4.3BSD" in the UNIX System Manager's Manual (SMM:12 and SMM:13) describe differences between 4.2BSD and 4.3BSD.
- Most user programs compiled and linked on 4.2/RT will execute correctly on 4.3/RT. Some do not, due to their use of kernel, networking, or system error logging facilities that differ in the two systems. Do not copy 4.2/RT system utilities into a 4.3/RT system. You must use the 4.3/RT versions.

New hardware supported:

- Addition of Extended ESDI disk support (See hd(4) for more information.)
- Addition of Small Computer System Interface (SCSI) adapter
- Support for the Advanced Processor Card (APC)
- Support for the MC68881 floating point chip on the APC

Kernel changes:

- In 4.2BSD, the directory /sys was a symbolic link to /usr/sys. This has been changed in 4.3BSD to a link to /usr/src/sys. 4.3/RT has not made this change, so that the directory /sys remains a symbolic link to /usr/sys.
- There have been changes to *namei* and related file system code which include 4.3 name caching and related changes for the Andrew File System (VICE). The version of the Andrew File System that ran on 4.2/RT will not run on 4.3/RT. An updated version is available from Carnegie Mellon University.
- There have been substantial 4.3 ttv changes.
- 4.3 include conventions have been followed in much of the kernel.
- Support for APC exception handling has been added, including exception stack and IOIM initialization.
- The minor device numbers of \(\frac{dev}{\ll tty,ms}\right) mono have changed. See the file \(\frac{dev}{MAKEDEV} \) for more information.
- Additional escape sequences have been added for the aed and ibmemul. See aed(4) and ibmemul(4) for more information.
- New screen switching routines have been added (color_table, color_entries, putstatus). Color parameters are now passed to screen_putc routines and screen blank routines. For more information, see "4.3/RT Console Emulators."

- The size of some of the mouse structures has been changed. *loctls* using the old size are still recognized for compatibility. See *xemul*(4) for more information.
- The programmable keyboard emulator has been changed to implement a true Caps Lock function, to allow 8-bit values in programmable keys, and to improve key click behavior. See *kbdemul*(4) for more information.
- The print screen function has been added to the *apa* devices. See *kbdemul*(4) for more information.
- Config file wildcarding for disks has been implemented. For example, this allows the system to run either an EESDI adapter with up to 3 disk drives or a system with up to 2 ESDI adapters and 3 disk drives. For more information, see "Building 4.3/RT Systems with Config."
- The contents of the *sigcontext* structure have been extended to include all of the machine-state at the time of the exception. This includes:
 - 1) 16 general registers
 - 2) 3 system registers (IAR, MQ, ICSCS). (Note that ICS and CS are separate registers that are stored as a single 32-bit quantity during interrupts.)
 - 3) Exception information (a count plus up to two 4-word exception packets)

In addition the sc_onstack word has been changed to be a 32-bit word of flags, with single bits to indicate the following:

SC_ONSTACK
SC_EXCEPTION

if executing on the signal stack exception information stored

SC_FLOATSAVE if floating point information was saved (SIGFPE floating point exceptions)

SC_FLOATSAVE is NOT set when the SIGFPE is an integer divide (FP_INT_DIVIDE).

See sigvec(2) for more information.

- The sigframe that previously contained a "floating point machine" image which was fpa/emulator specific now contains a "struct floatsave," which is used by all floating point support.
- Shared DMA support has been added. For more information, see "The DMA Reference Manual."
- The "kluster" paging facility, which attempts to page groups of pages to improve paging performance, is now available.
- User access to internal kernel segments (2-8) is now disabled.
- A higher clock rate is used for kernel profiling (512 ticks/sec) instead of the normal rate (64 ticks/sec).
- The delay loop parameter is adjusted according to actual CPU speed.
- Please note that in 4.3BSD the SIOCSPGRP *ioctl* now uses a positive argument as a process number and a negative argument as a process group number. See "socket" in Section 2 of "Bug Fixes and Changes in 4.3BSD" in the UNIX System Manager's Manual.

Driver changes:

• The token ring driver was updated to incorporate an ARP (Address Resolution Protocol) header change in order to be compatible with AIX. To facilitate conversion to the new header format, each token ring card is configurable to either the

- 4.2/RT token ring header format or the new format. The default configuration in this release is the format compatible with 4.2/RT. To configure the token ring to the updated format, see the **bridge** and **snap** options in *ifconfig*(8c).
- The streaming tape device driver has been improved and now includes support for multiple files on a single tape and backspacing by record. For more information, see st(4).
- Macros are now used to reference I/O devices and I/O memory.
- The disk driver now supports the Extended ESDI, with support for one-to-one interleave factor, simultaneous operation of multiple drives on one adapter, DMA, improved bad block handling, and the simultaneous operation of two adapters. See disk(4) and hd(4) for more information.
- Device driver interrupt routines now receive the IRQ (8259 interrupt level) as their second parameter. For more information, see "Building 4.3/RT Systems with Config."
- Device driver probe routines now have an additional parameter: a pointer to the controller structure. For more information, see "Building 4.3/RT Systems with Config."
- The autoconfiguration code in the diskette driver has been improved.

Standalone changes:

- Changes have been made to format(8) to format Extended ESDI and to convert from hd70r to hd70e. See format(8) and "IBM RT PC New Model Series Upgrade Instructions."
- Two 1M memory cards are no longer supported.
- Standalone code is now loaded at 1M, which allows kernels to be up to 1M text plus data.
- The standalone debugger has been improved to include implementation of screen display mode, better help messages, additional commands, and better traceback code. For more information, see debug(8).
- /sys/standca/debug now picks up termcap parameters so that it can run on non-console terminals and in X windows. For more information, see debug(8).

Utility changes:

- If a remote file system is used, commands that descend recursively through directories (particularly starting at /) may take a long time to execute. There are two find(1) commands, in /usr/adm/daily and /usr/lib/find/updatedb, that may be modified to avoid descending into remote file systems. For more information, see find(1).
- The High C compiler, hc, is the default C compiler obtained via cc. The portable C compiler is available via the pcc(1) command. The cc(1) man page describes how to change the default selection.

The High C compiler compiles faster than the portable C compiler, generates code that is generally shorter and faster, and supports a language level close to the proposed ANSI C standard. Preprocessor variables <u>STDC</u> and <u>HIGHC</u> permit conditional use of language features that differ between the compilers.

The High C compiler's default level of error-checking has been adjusted to be closer to the portable C compiler's, and the default size of the ENUM type has been changed to 4 bytes, for compatibility with the portable C compiler.

- For more information, see the hc(1) man page, the article "Recompiling with High C," and Appendix C, "The High C Programmer's Guide."
- The dbx source-language debugger has been upgraded and works with both hc and pcc. Programs compiled on a previous release must be recompiled in order to use 4.3/RT dbx.
- The versions of wm and BE1 distributed by Carnegic Mellon University which ran on 4.2/RT will not run on 4.3/RT. An updated version is available from Carnegie Mellon University.

Floating point changes:

• 4.3/RT supports two different floating-point engines: the Floating Point Accelerator (FPA); and the MC68881, which is standard on the Advanced Processor Card in Models 115 and 125. Floating-point software support in 4.3/RT is significantly different from 4.2/RT; please read the article "Floating Point Arithmetic" for details. Most users will be unaffected by these differences; most old programs will run without recompilation or relinking. To take advantage of the new floating point hardware however, programs must be recompiled.

2. DOCUMENTATION CHANGES

2.1. MANUAL PAGES

Various technical changes were made throughout the manual pages to support the new processor card and other hardware. All pages previously at the 4.2BSD level were revised to the 4.3BSD level, where appropriate.

Some manual pages describe functions that are at the 4.2/RT level and 4.2/RT utilities with 4.3BSD functionality added. For a list of these manual pages, see "Appendix A. Software Description" in Volume II.

The previous release of this product was at the 4.2BSD level but contained many 4.3BSD level functions. As a result, the previous edition of this manual contained 4.3BSD level pages because the user was required only to have access to 4.2BSD documentation. Because the product is now at the 4.3BSD level and the user is required to have access to 4.3BSD documentation, the manual pages previously included for some 4.3BSD functions have been removed.

The following pages were added for this release. The pages marked with an asterisk (*) are 4.3/RT only commands and function and do not appear in the 4.3BSD manuals.

- getfloatstate(2)*
- sc(4)*
- dbx(5)
- ifconfig(8c)
- newvd(8)*
- ptfinstall(8)*
- restore(8)
- restore.tape(8r)*
- rvdcopy(8)*
- rvdhosts(8)*
- scsiformat(8c)*
- sendapar(8)*
- syscall(8r)*
- tailor(8)*
- tbuffer(8r)*
- vdabort(8)*

2.2. SUPPLEMENTARY DOCUMENTS The following documents are new:

- IBM RT PC New Model Series Upgrade Instructions, which describes how to install the upgrade kit used to convert an IBM RT PC Model 15 or 25 to a Model 115 or 125.
- DMA Reference Manual, which describes the kernel routines 4.3/RT provides to device drivers that use the IBM RT PC Direct Memory Access (DMA) channels.

The following documents are revised for this release:

- Installing and Operating Academic Information Systems 4.3
- Building 4.3/RT Systems with Config
- Floating Point Arithmetic
- The IBM 3812 Pageprinter
- 4.3/RT Linkage Convention
- 4.3/RT Console Emulators
- The Remote Virtual Disk System

6

DECEMBER 15, 1986

1. TECHNICAL HIGHLIGHTS

The MetaWare High C Compiler

The High C compiler significantly enhances the performance of the system.

IBM RT PC 4Mb Memory Expansion

Use of this card with other expansion cards gives the user up to 8Mb of addressable random access memory.

Support for CMU-supplied Andrew (Vice/Virtue hooks)

Andrew consists of a prototype distributed file system and a window-based user interface that provides easy access to the power of the IBM RT PC, and a campus-wide local area network (LAN).

Updated support to allow 1.2Mb diskettes to be bootable

Support for X, Version 10, hooks

Support is included to allow use of the X, Version 10 modules from the 4.3 Berkeley Software Distribution (4.3BSD) release tape.

Remote Virtual Disk (RVD)

RVD allows sharing of all or part of a physical disk among a number of workstations over a Local Area Network (LAN).

All-Points-Addressable (APA) Screen Print Facility

A text or graphics image on a bitmap display can be captured and printed on an IBM 3812 Pageprinter, an IBM 4201 Proprinter, or an IBM 5152 Graphics Printer.

Optional MetaWare Professional Pascal Compiler

Professional Pascal was designed to make professional programming easier. It supports ANSI Standard Pascal and has many extensions. It is separately orderable.

Selected 4.3BSD Utilities

The following descriptions were extracted from "Bug Fixes and Changes in 4.3BSD," by M. K. McKusick, et al., dated April 15, 1986, in the 4.3BSD System Manager's Manual.

The user can now choose to run sh or csh. Mail can now be sent to the user after the job has run; mail is always sent if there were any errors during execution. At now runs with the user's full permissions. All spool files are now owned by "daemon." The last update time is in seconds instead of hours. The problems with day and year increments are fixed.

awk Problems when writing to pipes are corrected.

Problems opening standard input multiple times are fixed. Cat now runs much faster in the default (optionless) case.

checknr The .T& tbl directive was added to the list of known commands.

chgrp An option was added for recursively changing the group of a directory tree.

chmod Can now recursively modify the permissions on a directory tree. The mode string was extended to turn on the execute bit conditionally if the file is executable or is a directory.

clear Now has a proper exit status.

compress Replaces compact as the preferred method to use in saving file system space.

No longer suffers problems when copying a directory to a nonexistent name or when some directories are not writable in a recursive copy. The -p flag was added to preserve modes and times when copying files.

crypt Waits for makekey to finish before reading from its pipe.

ctags was modified to recognize LEX and YACC input files. Files ending in .y are presumed to be YACC input, and a tag is generated for each non-terminal defined, plus a tag yyparse for the first %% line in the file. Files ending in .l are checked to see if they are LEX or Lisp files. A tag yylex is generated for the first %% line in a LEX file. In addition, for both kinds of files, any C source after a second %% is scanned for tags.

dd Exit codes were changed to correspond with normal conventions.

deroff no longer throws out two letter words.

diff Context diffs merge nearby changes. New flags were added for ignoring white space differences and for insensitivity to case.

echo No longer accepts -nanything in place of -n.

error Support for the DEC Western Research Labs Modula-2 compiler was added.

from An error message is printed if the requested mailbox cannot be opened.

hostid Was extended to take an Internet address or hostname.

kill Signal 0 can now be used as documented.

lex The error messages were made more informative.

In Now prints a more accurate error message when asked to make a symbolic link into an unwritable directory.

lock Lock now has a default fifteen minute timeout. The root password can be used to override the lock. If an EOF is typed, it is now cleared instead of spinning in a tight loop until the timeout period.

mail Mail now expects RFC822 headers instead of the obsolete RFC733 headers. A retain command was added. If the PAGER variable is set in the environment, it is used to page messages instead of more(1). The write command now deletes the entire header instead of only the first line. An unread/Unread command (to mark messages as not read) was added. If Replyall is set, the senses of reply and Reply are reversed. When editing a different file, mail always prints the headers of the first few messages. Flock(2) is used for mailbox locking. Commands "-" and "+" skip over deleted messages; type user now does a substring match instead of a literal comparison. A -I flag was added which causes mail to assume input is a terminal.

Make A bug which caused *make* to run out of file descriptors because too many files and directories were left open is now fixed. Long path names should not be a problem now. A VPATH macro was added to allow the user to specify a path of directories to search for source files.

Support for alternate manual directories for man, apropos and whatis was added. A side effect of this is the whatis database was moved to the man directory. If the source for a manual page is not available, man will display the formatted version. This allows machines to avoid storing both formatted and unformatted versions of the manual pages. The environment variable MANPATH overrides the default directory |usr|man. The -t option is no longer supported. The printing process was streamlined by using "more -s catfile" instead of "cat -s catfile | ul | more -f". Searches of |usr|man|mano are more lenient about file name extensions. The

rm

source for man was considerably cleaned up; the magic search lists and commands were put at the top of the source file and the private copy of system was deleted.

mkdir Prints a "usage" error message instead of an uninformative "arg count" message.

more Now allows backward scanning. It will also handle window size changes. It simulates "crt" style erase and kill processing if the terminal mode includes those

options.

mv No longer runs cp(1) to copy a file; instead it does the copy itself.

nice Is relative as documented, not absolute.

pr The buffer is now large enough for 66 x 132 output.

ptx Cleans up after itself and exits with a zero status on successful completion.

quota Verifies the system supports quotas before trying to interpret the quota files.

ranlib The -t option updates a library's internal time stamp without rebuilding the table of contents. "Old format" and "mangled string table" are now warnings rather

than fatal errors. Memory allocation is done dynamically.

The key letter code was fixed so control characters are not generated. Several problems causing the generation of duplicate citations, particularly with the -e and -s options, were fixed. EOF on standard input is now properly handled. *Refer* folds upper and lower case when sorting.

The -f option produces no error messages and exits with status 0. The problem of

running out of file descriptors when doing a recursive remove was fixed.

rmdir Improved error messages, in the same fashion as mkdir.

size Now exits with the number of errors encountered.

sort Checks for and exits on write errors.

symorder Now reorders the string table as well as the name list.

tail Makes use of a much larger buffer.

Preserves modified times of extracted directories. The -B option is turned on when reading from standard input. Some sections were rewritten for efficiency.

tbl The hardwired line length was removed.

tee Tee's buffer size was increased.

Lock files are no longer left lying about after *tip* exits, and the *uucp* spool directory does not need to be world writable. A new "-\$" command sends output from a local program to a remote host. Alternate phone numbers are separated only by ","; thus several dialer characters previously illegal can now be used. *Tip* now arranges to copy a phone number argument to a safe place, then zero out the original version. This narrows the window in which the phone number is visible to miscreants using *ps* or *w*. Also fixed was a bug causing the phone number to be written in place of the connection message. Carrier loss is recognized and an appropriate disconnect action is taken. Bugs in calculating time and fielding signals are fixed. Several new dialers were added.

users Now much quieter if there are no users logged on.

vacation A new program answers mail while you are on vacation.

vgrind Extended to handle the DEC Western Research Labs Modula-2 compiler and yacc.

whereis Now also checks manl, mann, and mano.

whoami Uses the effective user id instead of the real user id.

xsend

Notice of secret mail is now sent with a subject line showing who sent the mail. The body of the message includes the name of the machine on which the mail can be read.

xstr

Now handles multiple-line strings.

Old Calling Sequence is no longer supported

The distributed kernel does not support the calling sequence used in versions of 4.2/RT distributed prior to March 1986. For more information, see "Programmer's Notes" in Volume II, Supplementary Documents.

2. DOCUMENTATION CHANGES

The December 15, 1986, release is a complete documentation revision.

Some manual pages from 4.3BSD were added to 4.2/RT. These manual pages are noted in their respective sections.

2.1. MANUAL PAGES

Various minor typographical and technical changes were made. Manual pages from "Appendix B. IBM Tools" were moved to Section 1 or Section 8. Other changes are noted below.

2.1.1. Section 1

In adb, the f option under \$modifier was added.

At, atq, and atrm are manual pages added from 4.3BSD; atq and atrm are new. They deal with the at command.

Bitprt is a new manual page describing the capture of images on a bitmap display.

Cat is a manual page added from 4.3BSD.

Cc is a revised manual page describing how High C (hc) or portable C (pcc) becomes the default compiler.

Chgrp and chmod are manual pages added from 4.3BSD.

Colpro was in Appendix B.

Compress is a new manual page added from 4.3BSD describing how to compress and expand data.

Cp, ctags, and dd are manual pages added from 4.3BSD.

Dosread was in Appendix B. It contains two new switches.

Dumpaed, dumpapa16, dumpapa8 and dumpapa8c are new manual pages describing how to dump their respective display's memories as binary files.

Error is a manual page added from 4.3BSD.

Hc is a new manual page describing the High C compiler.

Hostid is a manual page added from 4.3BSD.

Kbdlock was originally secure(i) in Appendix B.

Ld is updated with information about trampoline code.

Learn, leave, In, lock, mail, make, man, and more are manual pages added from 4.3BSD.

Pcc was originally cc. It still describes the pcc-based C compiler.

Pf and pic were in Appendix B.

Pp is a new manual page describing the Professional Pascal compiler. It is available only as an option of Professional Pascal.

Pprint, prfl, proff, and ptroff were in Appendix B.

Scale is a new manual page describing how to resize a bitmap image.

Support is a new manual page describing how to obtain support information for IBM RT PC hardware and software.

Telnet(1C) and tip(1C) are manual pages added from 4.3BSD.

Unifdef is a new manual page added from 4.3BSD describing how to remove ifdef'ed lines.

Up is a new manual page describing the up/down functions of the RVD utilities.

Vacation is a new manual page added from 4.3BSD describing the vacation function.

Vgrind is a manual page added from 4.3BSD describing how to grind nice listings of programs for the IBM 3812 Pageprinter.

Write is a manual page added from 4.3BSD.

2.1.2. Section 2

Intro was added from 4.2BSD. Error 70 (EBDBAD) was added.

Ptrace is a manual page from 4.3BSD describing the process trace. It was revised with IBM RT PC modifications.

Vdspin and vdstats are new manual pages describing the spin up or spindown of a remote virtual disk and how to acquire client RVD statistics respectively.

2.1.3. Section 3

Ctype, mktemp, and nlist are manual pages added from 4.3BSD.

Asinh(3M), hypot(3M), sin(3M), sinh(3M), and sqrt(3M) are updated with "Errors" sections.

Intro(3S) is a new manual page added from 4.3BSD. It is stdio in 4.3BSD.

Ferror(3S), fopen(3S), fseek(3S), getc(3S), and gets(3S) are manual pages added from 4.3BSD.

Scanf(3S) is updated with a "Notes" section.

Setbuf(3S) is a manual page added from 4.3BSD.

2.1.4. Section 4

Intro is updated with the new devices 4.2/RT supports as well as a section on emulators.

Aedemul is a new manual page describing the graphics interfaces for the IBM Academic Information Systems experimental display.

Bufemul is a new manual page describing the kernel buffering emulator.

Bus is a new manual page describing control of access to the system I/O bus.

Cons, disk, and fd are updated and clarified.

Ibm5151, ibm6153, ibm6154, ibm6155 and ibmaed are updated and clarified.

Ibmemul is a new manual page describing the IBM 3101 emulator.

Keyboard has had a name change. It is now kbdemul.

Lp has had the switch description updated and clarified.

Rvd(4P) is a new manual page describing the Remote Virtual Disk protocol.

Stdemul is a new manual page describing the standard output emulator.

Tb is a new manual page added from 4.3BSD describing the line discipline for displaying devices.

Tty added two line disciplines for the mouse.

Xemul is a new manual page describing the X input emulator for queuing keyboard and mouse events.

2.1.5. Section 5

Rvddb is a new manual page describing the RVD server configuration table.

Rvdtab is a new manual page describing information about client RVDs.

2.1.6. Section 8

Cvt38/2 and debug were in Appendix B.

Lpfilter(8R) is a new manual page describing the output filters for the IBM 4201 Proprinter and the IBM 5152 Graphics Printer.

Makedev has had displays, X, constty, mice, and floorstand added to the list of devices.

Makesym and omerge were in Appendix B.

Rvdchlog, rvddown, rvdexch, rvdflush, rvdgetm, rvdlog, rvdsend, rvdsetm, rvdshow, rvdshut, and rvdsrv are new manual pages describing functions of the RVD. Rvdchlog describes how to change logging levels; rvddown describes how to force spindown of an RVD; rvdexch describes how to exchange names of two remote RVDs; rvdflush describes how to spindown a client's RVD packs; rvdgetm describes how to get operations messages from an RVD server; rvdlog describes how to cause the RVD server to log statistics; rvdsend describes how to send a control stream to an RVD server; rvdsetm describes how set operations messages on an RVD server; rvdshow describes how to show connections to an RVD server; rvdshut describes how force a shutdown of an RVD server; and rvdsrv describes the RVD server daemon.

Savervd is a new manual page describing how to back up and restore RVD packs to and from tape.

Spinup is a new manual page describing how to spin up and down an RVD pack.

Vddb is a new manual page describing the RVD database manager.

Vdstats is a new manual page describing how to list client RVD statistics.

Width3812 was in Appendix B.

2.2. SUPPLEMENTARY DOCUMENTS

2.2.1. Operating 4.2/RT

"Hardware Supported" is updated.

2.2.2. Building 4.2/RT Systems with Config

"SECURE" is added to the optionlist in Section 4, "Configuration File Syntax."

/sys/cacons was added to the system source directories in Section 6, "Adding New System Software."

2.2.3. Assembler Reference Manual

The "Extended Mnemonics" tables were all made the same section level under Section 8, "Macro Instructions."

2.2.4. Floating Point Arithmetic

New information on the -lfpa flag is added.

2.2.5. Programmer's Notes

New information on variable types of char is added under Section 3, "Character Type is Unsigned."

Section 7, "Old Calling Sequence is No longer Supported," is new.

2.2.6. 4.2/RT Linkage Convention

Section 14, "Addressability in Very Large Modules" is new.

2.2.7. Recompiling with High C

This article is new and provides guidance to C programmers who recompile existing programs with MetaWare High C.

2.2.8. Professional Pascal Differences

This article is new and points out the major differences between Berkeley Pascal and MetaWare Professional Pascal, an optional feature.

2.2.9. 4.2/RT Console Emulators

This article is new and explains the need for, and the design of, emulators for 4.2/RT.

2.2.10. The Remote Virtual Disk System

This article is new and describes a network service that provides a client computer with the appearance of removable-media disk drives and an unlimited number of disk packs.

2.2.11. Appendices

"Appendix A. Software Description" is updated.

"Appendix B. IBM Tools" is now deleted. The manual pages in this appendix were moved to Sections 1 and 8. "Appendix C. Graphics Manual Pages for the IBM Academic Information Systems Experimental Display" is now Appendix B.

"Appendix C. High C Programmer's Guide" is new.

SEPTEMBER 30, 1986

1. TECHNICAL HIGHLIGHTS

Support for the IBM RT PC Baseband Adapter for use with Ethernet

The new adapter can be used with or instead of network adapters previously announced. The previously announced restriction against using two network adapter cards in the same machine is now removed; you can install two IBM Token-Ring adapters, two Ethernet adapters or one of each.

Support for the 70Mb Enhanced Small Device Interface (ESDI) Fixed-Disk Drives and ESDI Magnetic Media Adapter

Support for the IBM 6154 Advanced Color Graphics Display

The *ibm6154*(4) driver is provided to use the IBM 6154 Advanced Color Graphics Display for console output similar to the capability provided for monochrome displays.

Nroff support for the IBM 4201 Proprinter

The proff(i) utility lets you generate near-letter-quality documents on a local printer. Take advantage of such features as font styles, underlining, centering and footnoting. Output can be sent to other printers or written to the standard output for further processing.

AIX co-residence and 4.2A disk partitions

New disk partition support allows the Advanced Interactive Executive operating system (AIX) and 4.2A to co-reside in the same workstation. Only one operating system can be in execution at a time. The AIX/Virtual Resource Manager (VRM) minidisk table stores the 4.2A partition information. The table also lets you create non-standard 4.2A partitions. Use this method to increase the amount of paging space at the expense of reducing file storage space.

Speaker enhancement

Speaker(4) provides a single tone interface to the console speaker. The tone is suitable for music, alarms or bells.

Straight line floating-point accelerator performance enhancements

The Portable C Compiler (cc(1)) and f77 are enhanced to generate in-line floating point instructions instead of function calls. Floating-point-intensive programs run up to twice as fast as those compiled with the old compiler.

Enhanced workstation security

You can now electronically lock and disable the console keyboard, preventing unauthorized use of the console. The system can be rebooted remotely while in this software-locked state. To regain access to the console keyboard, you must use the system unit's metal key.

The *stdio* library is now at the 4.3BSD level.

Slots are dynamically allocated for file pointers. Output on unbuffered files is now buffered within a call to printf or fputs for efficiency. Fseek now returns zero if it was successful. Fread and fwrite are rewritten to improve performance. Fgets, gets, and fputs are written in assembler for better performance. Line buffering now works on any file descriptor, not just stdout or stderr. Putc is implemented completely within a macro except when the buffer is full or when a newline is encountered in the output stream on a line-buffered file. Some sign-extension bugs with the return value of putc are fixed.

You can improve the performance of existing utilities by relinking them after installing the September 30, 1986, system update.

2. DOCUMENTATION CHANGES

2.1. MANUAL PAGES

Various minor typographical and technical changes were made.

2.1.1. Section 3

Malloc is the 4.3BSD version.

Sinh is updated.

2.1.2. Section 4

Intro is updated with material related to this section.

Disk and hd have changes regarding 4.2A/AIX co-residence and the new Enhanced Small Device Interface (ESDI) Magnetic Media Adapter.

Ibm6154 is a new manual page describing the new Advanced Color Graphics Display interface.

Keyboard has an addition for the new security feature described in secure(i).

Lan is completely revised to reflect the changes made in the IBM RT PC Token-Ring Adapter support.

Speaker is a new manual page describing the new speaker facility.

Un has a name change reflecting support for the IBM RT PC Baseband Adapter for use with Ethernet, also referred to as the IBM Ethernet Adapter.

2.1.3. Section 8

Landump is a new manual page describing the new command to dump the IBM RT PC Token-Ring Adapter.

Newfs is taken from 4.2BSD and modified to reflect the 4.2A/AIX co-residence. Sautil is also modified to reflect the 4.2A/AIX co-residence.

Minidisk is a new manual page describing the new minidisk maintenance utility.

2.1.4. Section i

Dosread has two new parameters for initializing blank diskettes and creating bootable diskettes.

Colpro, prfl and proff are new manual pages describing the new IBM 4201 Proprinter and IBM 5152 Graphics Printer support.

Secure is a new manual page describing the new keyboard security feature.

2.2. SUPPLEMENTARY DOCUMENTS

2.2.1. Operating Academic Information Systems 4.2

Chapter 1, which deals with support, is updated with information about the IBM Ethernet Adapter and the Enhanced Small Device Interface (ESDI) Magnetic Media Adapter.

A new chapter, Chapter 5, "AIX and 4.2A Co-residence," is added.

2.2.2. Building 4.2A Systems with Config

Various minor typographical and technical changes were made.

A complete list of the changed pages is found on the cover sheet of the September 30, 1986, update packet, entitled "How to Update Your Manual."

JUNE 30, 1986

1. TECHNICAL HIGHLIGHTS

Pprint(i) utility

The *pprint* utility provides advanced-function text printing on the IBM 3812 Pageprinter. *Pprint* supports automatic page headings, page rotation and a choice of fonts. Page scaling lets you print wide pages or print two page images per page. *Pprint* queues print requests through the 4.2 *lpr* spooling system.

2. DOCUMENTATION CHANGES

2.1. MANUAL PAGES

Various minor typographical and technical changes were made.

2.1.1. Section 1

Find is an updated manual page stating it does not follow symbolic links.

Mt is updated with references and known bugs.

2.1.2. Section 4

Ibm6153 is updated with new information about displayable pixels.

Ibmaed is updates with escape sequences.

Lp is updated with information on the IBM 3812 Pageprinter.

St is updated with known bugs.

2.1.3. Section 5

Font3812 is updated with new /dat font file information.

2.1.4. Section i

Cvt3812 is updated with information on the IBM 3812 Pageprinter fonts.

Width3812 is updated to make the description more clear.

Pic and ptroff are new manual pages dealing describing the new ditroff facility. Ditroff (device-independent troff) support for the IBM 3812 Pageprinter¹ is a separately-licensed feature of 4.2A. It is a modified version of the Documenter's Workbench.

Pprint is a new manual page describing advanced-function text printing on the IBM 3812 Pageprinter.

¹Hereinafter referred to as "ditroff".

2.2. SUPPLEMENTARY DOCUMENTS

2.2.1. Operating Academic Information Systems 4.2A

Various minor typographical and technical changes were made.

2.2.2. Building 4.2A Systems with Config

Various minor typographical and technical changes were made.

2.2.3. The IBM 3812 Pageprinter

This article is completely replaced. The replacement contains information on new printing procedures as well as the ditroff facility.

A complete list of the changed pages is found on the cover sheet of the June 30, 1986, update packet, entitled "How to Update Your Manual."

This page intentionally left blank.

ABOUT THIS MANUAL

1. PURPOSE AND AUDIENCE

This multi-volume manual is for programmers, system managers, and users experienced with the 4.3 Berkeley Software Distribution¹ version of UNIX operating systems. The manual describes IBM Academic Operating System 4.3,² which is a port of the 4.3BSD operating system and runs on the IBM RT PC and the IBM 6152 Academic System. The manual also describes differences between IBM/4.3 and 4.3BSD, and provides information to help the system manager configure and operate IBM/4.3.

2. SOFTWARE DESCRIPTION

IBM/4.3 is a multi-user, multi-tasking operating system. Features include a comprehensive command language, device-independent input/output, extensive communications facilities, a hierarchical file system, and program development tools. It supports a wide range of environments and application areas, including timesharing, batch processing, program development, and document preparation. Two C language compilers, a FORTRAN 77 language compiler, an assembler, and system source code are shipped with the system. The C compilers allow for migration to the IBM RT PC or IBM 6152 Academic System of applications developed for other UNIX operating systems.

IBM/4.3 includes hardware support for the IBM 6152 Academic System, IBM RT PC 6151 Models 10, 15, and 115 and IBM RT PC 6150 Models 20, 25, and 125 processors. For a complete list of all other hardware supported, such as displays, printers, and adapters, see the "Installing and Operating Academic Operating System 4.3" article in Volume II.

Major subsystems include:

- 4.3BSD hierarchical file system
- Extensive communications support, including:
 - an IBM Ethernet network
 - a Token-ring network
 - asynchronous communications
 - uucp (which copies files between systems running UNIX operating systems)
 - TCP/IP
- Programming tools
- Document preparation tools
- Electronic mail
- System resource accounting

Software support includes:

- Shared read-only code
- Demand-paged virtual memory

¹Hereinafter referred to as "4.3BSD."

²Hereinafter referred to as "IBM/4.3."

IBM/4.3 About This Manual

- C shell and Bourne shell command languages
- Modular utilities
- Background and foreground processing
- Use of the machine as a gateway
- pcc, the Portable C compiler
- hc, the MetaWare High C compiler
- Source files for 4.3BSD tools and utilities
- Binary files for supported 4.3BSD tools and utilities
- IEEE 754 floating point arithmetic and math library
- f77, a FORTRAN 77 compiler
- dbx, a source level debugger
- Ditroff (device-independent troff) support for the IBM 3812 Pageprinter

Optional features include:

• pp, the MetaWare Professional Pascal compiler

3. CONTENTS OF THIS MANUAL

This multi-volume manual contains:

Volume I. Manual Pages is for all readers. It contains revised manual pages for the UNIX User's Reference Manual (URM), the UNIX Programmer's Reference Manual (PRM), and the UNIX System Manager's Manual (SMM) as well as new pages for new commands, devices, and subroutines.

- Section 1. Commands and Application Programs describes publicly-accessible, general-use commands. These are additions to and changes of material found in the 4.3BSD URM.
- Section 2. System Calls describes system calls. These are additions to and changes of material found in the 4.3BSD PRM.
- Section 3. Subroutines describes functions in various libraries. These are additions to and changes of material found in the 4.3BSD PRM.
- Section 4. Special Files describes special files, related driver functions, and networking support. These are additions to and changes of material found in the 4.3BSD PRM.
- Section 5. File Formats and Conventions describes file formats and conventions. These are additions to and changes of material found in the 4.3BSD PRM.
- Section 8. Maintenance Commands and Procedures contains commands for system operation and maintenance. These are additions to and changes of material found in the 4.3BSD SMM.

Volume II. Supplementary Documents contains information about configuring and operating IBM/4.3, as well as information for the programmer. It contains new and revised material for the 4.3BSD UNIX System Manager's Manual (SMM) and the UNIX Programmer's Supplementary Documents (PSI). The organization of source for the IBM/4.3 manual is discussed in the README file in /usr/doc/ibmdoc.

• Installing and Operating Academic Operating System 4.3 describes how to install and operate the operating system.

IBM/4.3 About This Manual

• Building IBM/4.3 Systems with Config describes config, a tool used in building IBM/4.3 system images, and provides information for using config on the IBM RT PC or IBM 6152 Academic System.

- IBM RT PC New Model Series Upgrade Instructions describes how to install the upgrade kit used to convert an IBM RT PC Model 15 or 25 to a Model 115 or 125.
- The IBM 3812 Pageprinter provides information for installing the IBM 3812 Pageprinter, and installing and converting fonts.
- IBM/4.3 Console Emulators explains the need for, and design of, emulators for IBM/4.3.
- The Remote Virtual Disk System describes a network service that provides a client computer with the appearance of removable-media disk drives and an unlimited number of disk packs.
- The DMA Reference Manual describes the utilities provided with the operating system for using the Direct Memory Access (DMA) channels.
- Assembler Reference Manual for IBM/4.3 describes the usage and input syntax of as, the IBM/4.3 assembler for the IBM RT PC and IBM 6152 Academic System. This article replaces the Assembler Reference Manual found in the 4.3BSD PS1.
- Floating Point Arithmetic summarizes floating point arithmetic in IBM/4.3.
- The C Subroutine Interface for the IBM Academic Information Systems Experimental Display describes an interface with graphics routines for the IBM Academic Information Systems experimental display.³
- Programmer's Notes is a brief compendium of insights, suggestions, and notes gathered from the programmers who ported applications to IBM/4.3.
- IBM/4.3 Linkage Convention describes the calling sequence used in 4.3 on the IBM RT PC and the IBM 6152 Academic System.
- Recompiling with High C provides guidance to C programmers who recompile existing programs with High C.
- Professional Pascal Differences points out the major differences between Berkeley Pascal and Professional Pascal as an aid to programmers who recompile existing programs with Professional Pascal.
- PIC -- A Graphics Language for Typesetting User Manual describes how to use PIC, a language used to draw simple figures on a typesetter.
- Appendix A. Software Description contains a list of 4.3BSD and IBM/4.3 functions supported in this distribution, as well as a list of unsupported functions.
- Appendix B. Graphics Manual Pages contains manual pages for the graphics routines provided with the C Subroutine Interface for the experimental display.
- Appendix C. High C Programmer's Guide contains a guide to programming with High C on the IBM RT PC and the IBM 6152 Academic System.

³Hereinafter referred to as "the experimental display."

IBM/4.3 About This Manual

4. HOW TO USE THIS MANUAL

Use the following table to locate information appropriate to your use of IBM/4.3.

User Type	Volume	Section or Article
	I	Sections 1 and 8
System Manager	II	Installing and Operating Academic Operating System 4.3 Building IBM/4.3 Systems with Config IBM RT PC New Model Series Upgrade Instructions The IBM 3812 Pageprinter IBM/4.3 Console Emulators The Remote Virtual Disk System The DMA Reference Manual Programmer's Notes Appendix A: Software Description
Programmer	I	All sections
	II	Assembler Reference Manual for IBM/4.3 Floating Point Arithmetic The C Subroutine Interface for the IBM Academic Information Systems Experimental Display Programmer's Notes IBM/4.3 Linkage Convention Recompiling with High C Professional Pascal Differences The Remote Virtual Disk System The DMA Reference Manual PIC A Graphics Language for Typesetting User Manual Appendix B: Graphics Manual Pages Appendix C: High C Programmer's Guide
User	I	Section 1

5. RELATED PUBLICATIONS

To use IBM/4.3, you should have access to the following publications:

- The 4.3BSD library:
 - UNIX User's Reference Manual (URM)
 - UNIX Programmer's Reference Manual (PRM)
 - UNIX User's Supplementary Documents (USD)
 - UNIX Programmer's Supplementary Documents, Volume 1 (PS1)
 - UNIX Programmer's Supplementary Documents, Volume 2 (PS2)
 - UNIX System Manager's Manual (SMM)
 - User Contributed Software (UCS)
- IBM RT PC Site Preparation Guide, GA23-1058
- IBM RT PC User Setup Guide, SV21-8020
- IBM RT PC Guide to Operations, SV21-8021

- IBM RT PC Problem Determination Guide, SV21-8022
- IBM RT PC Hardware Technical Reference, SV21-8024
- IBM RT PC 6150 System Unit Hardware Maintenance and Service, SV21-8025
- IBM RT PC 6151 System Unit Hardware Maintenance and Service, SV21-8026
- High C Language Reference Manual, from MetaWare Incorporated, Santa Cruz, CA, 1986
- High C Language Extensions Manual with Rationale & Tutorials, ibid, 1986
- Professional Pascal Documentation Set, ibid, 1986
- IBM Personal System/2 and Personal Computer BIOS Interface Technical Reference, 68X2260
- IBM Personal System/2 Model 50 and 60 Technical Reference, 68X2224

IBM/4.3 About This Manual

This page intentionally left blank.

VOLUME I. MANUAL PAGES

This volume contains revised manual pages for several commands, subroutines, and functions in the 4.3BSD URM, PRM, and SMM. It also contains pages for commands, subroutines, and functions specifically for IBM/4.3.

In the following table, manual pages marked with an asterisk (*) are IBM/4.3 functions not found in 4.3BSD. Manual pages marked with a section symbol (§) are at a level earlier than 4.3BSD.

Section	Revised or New Manual Pages
1	adb§, aedjournal*, aedrunner*, andrew*, as§, bitprt*, cc, colpro*, date, dbx, dosread*, dumpaed*, dumpapa16*, dumpapa8*, dumpapa8c*, error, f77§, hc*, kbdlock*, ld§, mset, mt, pcc§, pf*, pic*, pp*, pprint*, prfl*, proff*, ptroff*, scale*, support*, tar, tn3270, up*, vgrind, vmstat, Xibm*, xwindows*
2	intro, getfloatstate*, getfpemulator*, ptrace, sigvec, vdspin*, vdstats*
3	intro, abort, asinh(3M), atof, byteorder(3N), ctype, ecvt, exp(3M), floor(3M), fpa(3X)*, frexp*, hypot(3M), ieee*, intro(3X), j0(3M), lgamma(3M), malloc, math(3M), printf(3S), scanf(3S), signal(3C), sin(3M), sinh(3M), sqrt(3M)
3F	intro(3F) \S , abort(3F) \S , bessel(3F) \S , perror(3F) \S , rand(3F) \S , traper(3F) \S , trapov(3F) \S , trpfpe(3F) \S
4	intro, aedemul*, ap*, asy*, autoconf, bufemul*, bus*, cons, disk*, fd*, hd*, ibm5081*, ibm5151*, ibm5154*, ibm6153*, ibm6154*, ibm6155*, ibm8514*, ibmaed*, ibmemul*, kbdemul*, lan*, lp, mem, mouse*, mtio, psp*, rvd(4P)*, sc*, speaker*, st*, stdemul*, tb, tty, un, vga*, xemul*
5	a.out, consoles*, core, dbx, font3812*, keyboard_codes*, map3270, printer3812*, rc.config, rvddb*, rvdtab*
8	intro, aedtest*, afpacode(8R)*, badsect, config§, crash(8R), cvt3812*, cvtsym*, debug*, diskpart*, fdformat(8R)*, fdisk*, flcopy(8R)*, format(8R), halt, ibm3812pp*, ifconfig(8C), init, landump(8R)*, lpfilter(8R)*, makedev, makesym*, minidisk(8R)*, newfs, newvd*, omerge*, ppt*, pstat, ptfinstall*, reboot, restore, restore.tape*, rvdchlog*, rvdcopy*, rvddown*, rvdexch*, rvdflush*, rvdgetm*, rvdhosts*, rvdlog*, rvdsend*, rvdsetm*, rvdshow*, rvdshut*, rvdsrv*, sautil(8R)*, savervd*, scsiformat(8C)*, sendapar*, setid, setscreen*, spinup*, syscall*, syslogd, tailor*, tbuffer*, vdabort*, vddb*, vdstats*, width3812*

IBM/4.3 Manual Pages

This page intentionally left blank.

PERMUTED INDEX (REVISED/NEW PAGES)

	@: arithmetic on shell variables	csh(1)
xclock - X Window System, analog	/ digital clock	xclock(1)
lib2648: subroutines for the HP ibmemul: IBM	2648 graphics terminal	lib2648(3X)
mset: retrieve ASCII to IBM	3101 emulator	ibmemul(4) mset(1)
data base for mapping ASCII keystrokes into IBM	3270 keys. map3270:	map3270(5)
/cvt20to12, cvt00to12: convert IBM 3820 and IBM	3800 fonts for use with the IBM 3812 Pageprinter	cvt3812(8)
font3812: font structures for IBM 3820 and IBM 3800 fonts for use with the IBM	3812 fonts	font3812(5) cvt3812(8)
pprint: print text files on IBM	3812 Pageprinter	pprint(1)
ppt: spooling system filter for the IBM	3812 Pageprinter	ppt(8)
ptroff: print troff files on IBM vgrind: grind nice listings of programs for the IBM	3812 Pageprinter	ptroff(1)
width3812: build width tables for IBM	3812 Pageprinter fonts	vgrind(1) width3812(8)
ibm3812pp: IBM	3812 Pageprinter server	ibm3812pp(8)
printer3812: IBM	3812 Pageprinter status information	printer3812(5)
cvt3812, cvt20to12, cvt00to12: convert IBM mtio: 4.	3820 and IBM 3800 fonts for use with the IBM 3812/ 3/RT magtape interface	cvt3812(8) mtio(4)
openpl et al.: f77 library interface to plot	(3X) libraries plot:	plot(3F)
colpro: column filter for IBM	4201 Proprinter	colpro(1)
ibmbit, ibmgra, ibmpro: output filters for the IBM proff: nroff for the IBM	4201 Proprinter and IBM 5152 Graphics Printer 4201 Proprinter and IBM 5152 Graphics Printer	lpfilter(8r) proff(1)
post-processing filter. prfl: IBM	4201 Proprinter and 15M 5152 Graphics Printer	prfl(1)
mtio:	4.3/RT magtape interface	mtio(4)
ibm5081, mpel – IBM ibm5151, mono: IBM	5081 Mega Pel Display interface	ibm5081(4) ibm5151(4)
output filters for the IBM 4201 Proprinter and IBM	5152 Graphics Printer. ibmbit, ibmgra, ibmpro:	lpfilter(8r)
proff: nroff for the IBM 4201 Proprinter and IBM	5152 Graphics Printer	proff(1)
prfl: IBM 4201 Proprinter/IBM	5152 Graphics Printer nroff post-processing filter	prfl(1)
ibm5154, ega: IBM interface. ibm6153, apa8: IBM	5154 Enhanced Graphics Display interface 6153 Advanced Monochrome Graphics Display	ibm5154(4) ibm6153(4)
ibm6154, apa8c: IBM	6154 Advanced Color Graphics Display interface	ibm6154(4)
interface. ibm6155, apa16: IBM	6155 Extended Monochrome Graphics Display	ibm6155(4)
ibm8514, ibm8604. ibm8514: IBM scsiformat: format the IBM	8514/A Display adapter for the ibm8503, ibm8513, 9332 disk unit	ibm8514(4) scsiformat(8c)
Interface (SCSI) Adapter. sc: IBM	9332 disks using the IBM Small Computer System	sc(4)
vdabort:	abort and spin down a drive	vdabort(8)
	abort: generate a fault.	abort(3)
abort: terminate	abort: terminate abruptly with memory image abruptly with memory image	abort(3F) abort(3F)
	abs: integer absolute value.	abs(3)
abs: integer	absolute value.	abs(3)
hypot, cabs: Euclidean distance, complex fabs, floor, ceil:	absolute value	hypot(3M) floor(3M)
aedemul: graphics interfaces for the IBM	Academic Information Systems experimental display.	aedemul(4)
interface. ibmaed, aed: IBM	Academic Information Systems experimental display .	ibmaed(4)
self-tests. aedtest: IBM test, and bring online the Advanced Floating Point	Academic Information Systems experimental display Accelerator. afpacode: load,	aedtest(8) afpacode(8r)
fpa: direct interface to floating point	accelerator	fpa(3X)
setscreen: control display screen	access	setscreen(8)
xhost - X window system	access control program.	xhost(1)
initgroups: initialize group	access: determine accessibility of a file	access(3F) initgroups(3)
bus: control of	access to the system I/O bus.	bus(4)
access: determine	accessibility of a file.	access(3F)
their inverses. sin, cos, tan, asin, asinh,	acos, atan, atan2: trigonometric functions and acosh, atanh: inverse hyperbolic functions.	sin(3M) asinh(3M)
statistics. vdstats:	acquire client Remote Virtual Disk (RVD)	vdstats(2)
signal: change the	action for a signal	signal(3F)
lan: IBM RT PC Token-Ring landump: dump IBM Token-Ring Personal Computer	Adapter	lan(4) landump(8r)
the IBM Small Computer System Interface (SCSI)	Adapter. sc: IBM 9332 disks using	sc(4)
ibm8514: IBM 8514/A Display	adapter for the ibm8503, ibm8513, ibm8514, ibm8604	ibm8514(4)
un: IBM RT PC Baseband	Adapter for use with Ethernet	un(4)
ns_addr, ns_ntoa: Xerox NS(tm)	add: debugger	adb(1) ns(3N)
inet_makeaddr, inet_lnaof, inet_netof: Internet	address manipulation routines. /inet_ntoa,	inet(3N)
loc: return the	address of an object	loc(3F)
getfpemulator: return ibm6154, apa8c: IBM 6154	address of the floating-point emulator	getfpemulator(2) ibm6154(4)
afpacode: load, test, and bring online the	Advanced Floating Point Accelerator	afpacode(8r)
ibm6153, apa8: IBM 6153	Advanced Monochrome Graphics Display interface	ibm6153(4)

dumpaed: dump	aed display memory as a binary file.	dumpaed(1)
display interface. ibmaed, Information Systems experimental display.	aed: IBM Academic Information Systems experimental aedemul: graphics interfaces for the IBM Academic	ibmaed(4) aedemul(4)
imormation bystems experimental display.	aedjournal: display commands in a log file	aedjournal(1)
	aedrunner: execute graphics commands in a log file	aedrunner(1)
experimental display self-tests.	aedtest: IBM Academic Information Systems	aedtest(8)
Floating Point Accelerator.	afpacode: load, test, and bring online the Advanced	afpacode(8r)
mem, kmem, kmem1, kmem2, kmem4, ros,	afpamem: main memory	mem(4)
learn: computer	aided instruction about UNIX	learn(1)
	alarm: execute a subroutine after a specified time	alarm(3F) alarm(3C)
	alias: shell macros.	csh(1)
unalias: remove	aliases.	csh(1)
valloc:	aligned memory allocator.	valloc(3C)
malloc, free, realloc, calloc,	alloca: memory allocator	malloc(3)
malloc, free, realloc, calloc, alloca: memory	allocator	malloc(3)
malloc, free, falloc: memory	allocator.	malloc(3F)
valloc: aligned memory scandir,	allocator	valloc(3C) scandir(3)
limit:	alter per-process resource limitations.	csh(1)
else:	alternative commands	csh(1)
xclock - X Window System,	analog / digital clock	xclock(1)
error:	analyze and disperse compiler error messages	error(1)
discipline.	ap: asynchronous data mode protocol line	ap(4)
dumpapa16: dump	apa16 display memory as a binary file	dumpapa16(1)
Display interface. ibm6155,	apa16: IBM 6155 Extended Monochrome Graphics apa8 display memory as a binary file	ibm6155(4)
dumpapa8: dump interface. ibm6153,	apa8: IBM 6153 Advanced Monochrome Graphics Display	dumpapa8(1) ibm6153(4)
dumpapa8c: dump	apa8c display memory as a binary file	dumpapa8c(1)
interface. ibm6154,	apa8c: IBM 6154 Advanced Color Graphics Display	ibm6154(4)
sendapar: send	APAR	sendapar(8)
.PP uwm - Window Manager Client	Application of X.PP	uwm(1)
graphics/ plot: openpl, erase, label, line, circle,	arc, move, cont, point, linemod, space, closepl:	plot(3X)
tar: tape	archiver.	tar(1)
arff, floopy:	archiver and copier for floppy	arff(8V)
VI_Copy: copy an disk: format of reserved	areas of the hard disk.	copy(3G) disk(4)
disk. format of fescived	arff, floopy: archiver and copier for floppy.	arff(8V)
glob: filename expand	argument list.	csh(1)
shift: manipulate	argument list.	csh(1)
varargs: variable	argument list.	varargs(3)
echo: echo	arguments	csh(1)
getarg, iargo: return command line	arguments	getarg(3F)
getopt: get option letter from m_out, sdiv, itom: multiple precision integer	argy	getopt(3)
traper: trap	arithmetic errors.	mp(3X) traper(3F)
@:	arithmetic on shell variables.	csh(1)
dumpaed: dump aed display memory	as a binary file.	dumpaed(1)
dumpapa16: dump apa16 display memory	as a binary file	dumpapa16(1)
dumpapa8: dump apa8 display memory	as a binary file.	dumpapa8(1)
dumpapa8c: dump apa8c display memory		dumpapa8c(1)
gmtime, asctime, timezone: convert date and time to	as: assembler	as(1) ctime(3)
map3270: data base for mapping	ASCII keystrokes into IBM 3270 keys	map3270(5)
fdate: return date and time in an	ASCII string.	fdate(3F)
mset: retrieve	ASCII to IBM 3270 keyboard map	mset(1)
atof, atoi, atol: convert	ASCII to numbers	atof(3)
ctime, localtime, gmtime,	asctime, timezone: convert date and time to ASCII	ctime(3)
and their inverses. sin, cos, tan,	asin, acos, atan, atan2: trigonometric functions	sin(3M)
no:	asinh, acosh, atanh: inverse hyperbolic functions assembler	asinh(3M) as(1)
as: a.out:	assembler and link editor output.	a.out(5)
a. out.	assert: program verification.	assert(3)
setbuf, setbuffer, setlinebuf:	assign buffering to a stream.	setbuf(3S)
tailor: work station customizing	assistance	tailor(8)
interface.	asy: multi-port asynchronous communications RS232C .	asy(4)
asy: multi-port	asynchronous communications RS232C interface	asy(4)
ap:	asynchronous data mode protocol line discipline	ap(4)
inverses. sin, cos, tan, asin, acos,	atan, atan2: trigonometric functions and their	sin(3M)
sin, cos, tan, asin, acos, atan, asinh, acosh,	atan2: trigonometric functions and their inverses atanh: inverse hyperbolic functions	sin(3M) asinh(3M)
asimi, acosti,	atof, atoi, atol: convert ASCII to numbers.	atof(3)
atof,	atoi, atol: convert ASCII to numbers	atof(3)
atof, atoi,	atol: convert ASCII to numbers	atof(3)
code.	autoconf: diagnostics from the autoconfiguration	autoconf(4)
autoconf: diagnostics from the	autoconfiguration code	autoconff4)

		-1 4/15
xload - X window system load	average display	
bg: place job in wait: wait for	background processes to complete	csh(1) csh(1)
badsect: create files to contain	bad sectors.	
budseot. Or cute they to contain	badsect: create files to contain bad sectors.	
keys. map3270: data	base for mapping ASCII keystrokes into IBM 3270 .	
vddb: Remote Virtual Disk (RVD) data	base manager	vddb(8)
fetch, store, delete, firstkey, nextkey: data	base subroutines. dbminit,	dbm(3X)
dbm_nextkey, dbm_error, dbm_clearerr: data	base subroutines. /dbm_delete, dbm_firstkey,	ndbm(3)
un: IBM RT PC	Baseband Adapter for use with Ethernet	un(4)
xcalc: X	based scientific calculator	xcalc(1)
bcopy,	bcmp, bzero, ffs: bit and byte string operations	bstring(3)
operations.	bcopy, bcmp, bzero, ffs: bit and byte string	bstring(3)
file. VI_Login, VI_Logout:	begin logging subroutine calls and close a log	log(3G)
$\overline{j0}$, $j1$, jn , $y\overline{0}$, $y1$, yn :	Bessel functions.	j0(3M)
The state of the s	bessel functions: of two kinds for integer orders	bessel(3F)
random, drandm, irandm:	better random number generator	random(3F)
changing/ random, srandom, initstate, setstate:	better random number generator; routines for	random(3)
dumpaed: dump aed display memory as a	bg: place job in background	csh(1)
dumpapa16: dump apa16 display memory as a	binary file.	dumpaed(1) dumpapa16(1)
dumpaparo: dump aparo display memory as a	binary file.	dumpapa8(1)
dumpapa8c: dump apa8c display memory as a	binary file.	dumpapa8c(1)
fread, fwrite: buffered	binary input/output.	fread(3S)
bcopy, bcmp, bzero, ffs:	bit and byte string operations.	bstring(3)
functions.	bit: and, or, xor, not, rshift, lshift bitwise	bit(3F)
Tunctions.	bitmap: bitmap editor for X window system	bitmap(1)
bitprt: capture the image on a	bitmap display and print it on an IBM printer	bitprt(1)
bitmap:	bitmap editor for X window system	bitmap(1)
scale: resize a	bitmap image.	scale(1)
print it on an IBM printer.	bitprt: capture the image on a bitmap display and	bitprt(1)
bit: and, or, xor, not, rshift, lshift	bitwise functions.	bit(3F)
fdisk:	boot record partition table maintenance utility	fdisk(8)
reboot:	bootstrapping procedures	reboot(8)
switch: multi-way command	branch	csh(1)
•	break: exit while/foreach loop	csh(1)
	breaksw: exit from switch.	csh(1)
fg:	bring job into foreground	csh(1)
Accelerator. afpacode: load, test, and	bring online the Advanced Floating Point	afpacode(8r)
	busemul: kernel buffering emulator	bufemul(4)
fread, fwrite:	buffered binary input/output	fread(3S)
stdio: standard	buffered input/output package	stdio(3S)
tbuffer: streaming tape	buffered read.	tbuffer(8)
bufemul: kernel	buffering emulator	bufemul(4)
setbuf, setbuffer, setlinebuf: assign	buffering to a stream.	setbuf(3S)
config:	build system configuration files.	config(8)
width3812:	build width tables for IBM 3812 Pageprinter fonts	width3812(8)
bus: control of access to the system I/O	bus.	bus(4)
maker assume relices between heat and nationals	bus: control of access to the system I/O bus	bus(4)
ntohs: convert values between host and network		byteorder(3N)
bcopy, bcmp, bzero, ffs: bit and swab: swap	byte string operations	bstring(3) swab(3)
bcopy, bcmp,	bzero, ffs: bit and byte string operations.	bstring(3)
cc: default	C compiler	cc(1)
he: High	C compiler.	hc(1)
pcc: pcc-based	C compiler.	pcc(1)
Xlib:	C Language X Window System Interface Library	Xlib(3X)
intro: introduction to	C library functions.	intro(3)
hypot,	cabs: Euclidean distance, complex absolute value	hypot(3M)
diskpart:	calculate default disk partition sizes	diskpart(8)
xcalc: X based scientific	calculator	xcalc(1)
syscall: system	call interface program	syscall(8)
getuid, getgid: get user or group ID of the	caller	getuid(3F)
malloc, free, realloc,	calloc, alloca: memory allocator	malloc(3)
siginterrupt: allow signals to interrupt system	calls	siginterrupt(3)
VI_Login, VI_Logout: begin logging subroutine	calls and close a log file	log(3G)
intro: introduction to system	calls and error numbers	intro(2)
on an IBM printer. bitprt:	capture the image on a bitmap display and print it	bitprt(1)
XMenu - X Deck of	cards Menu System	XMenu(3X)
	case: selector in switch.	csh(1)
default:	catchall clause in switch.	csh(1)
statistics. rvdlog:	cause Remote Virtual Disk (RVD) server to log	rvdlog(8)
	cbrt, sqrt: cube root, square root.	sqrt(3M)
	cc: default C compiler	∞(1)
Catal C	cd: change directory	csh(1)
fabs, floor,	ceil: absolute value, floor, ceiling functions	floor(3M)

fabs, floor, ceil: absolute value, floor,	ceiling functions	floor(3M)
chdir:	change default directory	chdir(3F)
cd:	change directory.	csh(1)
chdir:	change directory	csh(1) ioinit(3F)
server. rvdchlog:	change logging level of Remote Virtual Disk (RVD)	rvdchlog(8)
chmod:	change mode of a file.	chmod(3F)
chmod, fchmod:	change mode of file.	chmod(2)
umask: chown, fchown:	change or display file creation mask	csh(1) chown(2)
VI Color:	change screen color.	color(3G)
signal:	change the action for a signal	signal(3F)
set:	change value of shell variable	csh(1)
better random number generator; routines for ungetc: push	changing generators. /srandom, initstate, setstate: character back into input stream	random(3) ungetc(3S)
iscntrl, isascii, toupper, tolower, toascii:	character classification macros. /isprint, isgraph,	ctype(3)
getc, fgetc: get a	character from a logical unit.	getc(3F)
index, rindex, Inblnk, len: tell about	character objects	index(3F)
getc, getchar, fgetc, getw: get	character or word from stream.	getc(3S)
putc, putchar, fputc, putw: put putc, fputc: write a	character or word on a stream	putc(3S) putc(3F)
pato, ipato. Witto u	chdir: change default directory.	chdir(3F)
	chdir: change directory	csh(1)
	chmod: change mode of a file	chmod(3F)
	chmod, fchmod: change mode of file	chmod(2) chown(2)
VI Circle: draw a	, , ,	circle(3G)
closepl:/ plot: openpl, erase, label, line,	circle, arc, move, cont, point, linemod, space,	plot(3X)
infinity,/ copysign, drem, logb, scalb, rint,	classdouble, classfloat, isnan, unordered, finite,	ieee(3)
copysign, drem, logb, scalb, rint, classdouble,	classfloat, isnan, unordered, finite, infinity,/	ieee(3)
isascii, toupper, tolower, toascii: character default: catchall	classification macros. /isprint, isgraph, isentrl, clause in switch	ctype(3) csh(1)
ferror, feof,	clearerr, fileno: stream status inquiries.	ferror(3S)
.PP uwm - Window Manager	Client Application of X.PP	uwm(1)
vdstats: acquire	client Remote Virtual Disk (RVD) statistics	vdstats(2)
vdstats: list	client Remote Virtual Disk (RVD) statistics	vdstats(8)
up, down: /etc/rvd/rvdtab: information about	client Remote Virtual Disk (RVD) utilities client Remote Virtual Disks (RVDs)	up(1) rvdtab(5)
rydflush: spindown	client's Remote Virtual Disk (RVD) packs	rvdflush(8)
VI_Clip: set	clipping window.	clip(3G)
xclock - X Window System, analog / digital	clock.	xclock(1)
VI_Logout: begin logging subroutine calls and fclose, fflush:	close a log file. VI_Login,	log(3G) fclose(3S)
opendir, readdir, telldir, seekdir, rewinddir,	closedir: directory operations.	directory(3)
syslog, openlog,	closelog, setlogmask: control system log	syslog(3)
circle, arc, move, cont, point, linemod, space,	closepl: graphics interface. /erase, label, line,	plot(3X)
autoconf: diagnostics from the autoconfiguration VI Color: change screen	code	autoconf(4) color(3G)
ibm6154, apa8c: IBM 6154 Advanced	Color Graphics Display interface	ibm6154(4)
ionivio i, upuo i ibili o io i ila imio i	colpro: column filter for IBM 4201 Proprinter	colpro(1)
colpro:	column filter for IBM 4201 Proprinter	colpro(1)
exec: overlay shell with specified	command.	csh(1)
time: time routines for returning a stream to a remote	command	csh(1) rcmd(3)
rexec: return stream to a remote	command.	rexec(3)
system: issue a shell	command	system(3)
system: execute a UNIX	command.	system(3F)
switch: multi-way rehash: recompute	command branch	csh(1) csh(1)
unhash: discard	command hash table.	csh(1)
hashstat: print	command hashing statistics	csh(1)
nohup: run	command immune to hangups.	csh(1)
getarg, iargc: return repeat: execute	command line arguments	getarg(3F) csh(1)
onintr: process interrupts in	command scripts.	csh(1)
goto:	command transfer	csh(i)
else: alternative	commands	csh(1)
introduction to system maintenance and operation while; repeat	commands. intro:	intro(8) csh(1)
source: read	commands from file.	csh(1)
aedjournal: display	commands in a log file	aedjournal(1)
aedrunner: execute graphics	commands in a log file.	aedrunner(1)
asy: multi-port asynchronous cc: default C	communications RS232C interface	asy(4)
f77: FORTRAN 77	compiler	∝(1) ſ77(1)
hc: High C	compiler	hc(1)

man man hared C	aamailaa	man(1)
pcc: pcc-based C	compiler	pcc(1)
pp: Professional Pascal	compiler	pp(1)
error: analyze and disperse	compiler error messages	error(1)
wait: wait for background processes to	complete	csh(1)
hypot, cabs: Euclidean distance,	complex absolute value	hypot(3M)
landump: dump IBM Token-Ring Personal	Computer Adapter	landump(8r)
learn:	computer aided instruction about UNIX	learn(1)
sc: IBM 9332 disks using the IBM Small	Computer System Interface (SCSI) Adapter	sc(4)
endif: terminate	conditional	csh(1)
if:	conditional statement.	csh(1)
		. 1.1
while: repeat commands	conditionally.	csh(1)
	config: build system configuration files	config(8)
rc.	config: configuration file for startup scripts	rc.config(5)
rc.config:	configuration file for startup scripts	rc.config(5)
config: build system	configuration files	config(8)
rvddb: Remote Virtual Disk (RVD) server	configuration table	rvddb(S)
ifconfig:	configure network interface parameters	ifconfig(8c)
rvdshow: show	connections to Remote Virtual Disk (RVD) server.	rvdshow(8)
	cons: keyboard and console display interface	cons(4)
cons: keyboard and	console display interface	cons(4)
speaker:	console speaker interface	speaker(4)
· · · · · ·	consoles: utility database of display screens	consoles(5)
newfs:	construct a new file system.	newfs(8)
		` '
vlimit: control maximum system resource	consumption.	vlimit(3C)
openpl, erase, label, line, circle, arc, move,	cont, point, linemod, space, closepl: graphics/	plot(3X)
badsect: create files to	contain bad sectors	badsect(8)
rvdcopy: copy	contents of one RVD disk pack to another	rvdcopy(8)
***************************************	continue: cycle in loop	csh(1)
setscreen:	control display screen access.	setscreen(8)
	· ·	
init: process	control initialization.	init(8)
vlimit:	control maximum system resource consumption	vlimit(3C)
bus:	control of access to the system I/O bus	bus(4)
xhost - X window system access	control program	xhost(1)
rvdsend - send	control stream to Remote Virtual Disk (RVD) server	rvdsend(8)
syslog, openlog, closelog, setlogmask:	control system log.	syslog(3)
VI_FDefnCur, VI_EnCur, VI_DisCur, VI_PosnCur:	control the display cursor. VI_MDefnCur,	cursor(3G)
ecvt, fcvt, gcvt: output	conversion	ecvt(3)
long, short: integer object	conversion	long(3F)
printf, fprintf, sprintf: formatted output	conversion	printf(3S)
scanf, fscanf, sscanf: formatted input	conversion	scanf(3S)
ns addr, ns ntoa: Xerox NS(tm) address	conversion routines	ns(3N)
atof, atoi, atol:	convert ASCII to numbers.	atof(3)
ctime, localtime, gmtime, asctime, timezone:	convert date and time to ASCII	ctime(3)
the IBM 3812/ cvt3812, cvt20to12, cvt00to12:	convert IBM 3820 and IBM 3800 fonts for use with .	cvt3812(8)
cvtsym:	convert symbol table	cvtsym(8)
htonl, htons, ntohl, ntohs:	convert values between host and network byte order	byteorder(3N)
flcopy:	copier for diskettes.	flcopy(8r)
	copier for floppy.	arff(8V)
arff, flcopy: archiver and		
VI_Copy:	copy an area.	
rvdcopy:	copy contents of one RVD disk pack to another	rvdcopy(8)
fork: create a	copy of this process	fork(3F)
remainder, exponent manipulations.	copysign, drem, finite, logb, scalb: copysign,	ieee(3M)
classfloat, isnan, unordered, finite, infinity,/	copysign, drem, logb, scalb, rint, classdouble,	ieee(3)
copysign, drem, finite, logb, scalb:	copysign, remainder, exponent manipulations.	ieee(3M)
nfabort: dump	core and log it in a notesfile.	nfabort(3)
	core: format of memory image file	core(5)
functions and their inverses. sin,	cos, tan, asin, acos, atan, atan2: trigonometric	sin(3M)
sinh,	cosh, tanh: hyperbolic functions	sinh(3M)
· ,	crash: what happens when the system crashes	crash(8r)
	crash: what happens when the system crashes	crash(8V)
crash: what happens when the system	crashes.	crash(8r)
crash: what happens when the system	crashes.	crash(8V)
fork:	create a copy of this process	fork(3F)
(RVD). newvd:	create a new filesystem on a Remote Virtual Disk	newvd(8)
badsect:	create files to contain bad sectors	badsect(8)
umask: change or display file	creation mask.	csh(1)
Timore origing of dishial inc	crypt, setkey, encrypt: DES encryption	crypt(3)
convert data and time to ACCII	ctime, localtime, gmtime, asctime, timezone:	
convert date and time to ASCII.		ctime(3)
time,	ctime, Itime, gmtime: return system time	time(3F)
cbrt, sqrt:	cube root, square root.	sqrt(3M)
hostnm: get name of	current host.	hostnm(3F)
jobs: print		csh(1)
VI_AMove, VI_RMove: move the	current job list	
petcwd net nethneme of	current point	move(3G)
getcwd: get pathname of	current point	move(3G) getcwd(3F)
getcwd: get pathname of getwd: get motion.	current point	move(3G) getcwd(3F) getwd(3)

VI_DisCur, VI_PosnCur: control the display	cursor. VI_MDefnCur, VI_FDefnCur, VI_EnCur,	cursor(3G)
curses: screen functions with "optimal"	cursor motion.	curses(3X)
spline: interpolate smooth	curve	spline(1G)
tailor: work station		tailor(8)
use with the IBM 3812/ cvt3812, cvt20to12,		cvt3812(8)
fonts for use with the IBM 3812/ cvt3812,		cvt3812(8)
IBM 3800 fonts for use with the IBM 3812/	cvt3812, cvt20to12, cvt00to12: convert IBM 3820 and .	cvt3812(8)
oontinus.	cvtsym: convert symbol table	cvtsym(8)
continue:	cycle in loop.	csh(1) rvdsrv(8)
rvdsrv: Remote Virtual Disk (RVD) server VI Dash: set line	daemon	dash(3G)
eval: re-evaluate shell	data	csh(1)
VI MRead, VI FRead: read display	data.	read(3G)
3270 keys. map3270:	data base for mapping ASCII keystrokes into IBM	map3270(5)
vddb: Remote Virtual Disk (RVD)	data base manager.	vddb(8)
dbminit, fetch, store, delete, firstkey, nextkey:	data base subroutines.	dbm(3X)
dbm firstkey, dbm nextkey, dbm error, dbm clearerr:	data base subroutines. /dbm store, dbm delete,	ndbm(3)
ap: asynchronous	data mode protocol line discipline	ap(4) `
consoles: utility	database of display screens	consoles(5)
xrdb - Server Resource	Database Utility	xrdb(1)
date: print and set the	date	date(1)
time, ftime: get		time(3C)
fdate: return	date and time in an ASCII string.	fdate(3F)
localtime, gmtime, asctime, timezone: convert	date and time to ASCII. ctime,	ctime(3)
idate, itime: return	date or time in numerical form.	idate(3F)
	date: print and set the date.	date(1)
dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error,	dbm_clearerr: data base subroutines. /dbm_store,	ndbm(3)
dbm_firstkey, dbm_nextkey, dbm_error,/ dbm_open,	dbm_close, dbm_fetch, dbm_store, dbm_delete,	ndbm(3)
dbm_open, dbm_close, dbm_fetch, dbm_store,	dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error,/	ndbm(3)
/dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey,	dbm_error, dbm_clearerr: data base subroutines	ndbm(3)
dbm_nextkey, dbm_error,/dbm_open, dbm_close,	dbm_fetch, dbm_store, dbm_delete, dbm_firstkey,	ndbm(3)
data/ /dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines.	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey:	ndbm(3) dbm(3X)
/dbm fetch, dbm store, dbm delete, dbm firstkey,	dbm nextkey, dbm error, dbm clearerr: data base/	ndbm(3)
dbm delete, dbm firstkey, dbm nextkey, dbm error,/	dbm_open, dbm_close, dbm_fetch, dbm_store,	ndbm(3)
dbm error,/ dbm open, dbm close, dbm fetch,	dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey,	ndbm(3)
doni_crror, doni_open, doni_crose, doni_recen,	dbx: dbx symbol table information.	dbx(5)
	dbx: debugger.	dbx(1)
dbx:		dbx(5)
	debug: debugger for the IBM RT PC	debug(8)
adb:	debugger	adb(1)
dbx:	debugger	dbx(1)
debug:	debugger for the IBM RT PC	debug(8)
makesym: make	debugger symbol table	makesym(8)
XMenu - X	Deck of cards Menu System	XMenu(3X)
cc:	default C compiler.	cc(1)
	default: catchall clause in switch	csh(1)
chdir: change	default directory.	chdir(3F)
diskpart: calculate	default disk partition sizes.	diskpart(8)
kbdemul:	default keyboard emulator	kbdemul(4)
stty, gtty: set and get terminal state	(defunct).	stty(3C)
dbminit, fetch, store,	delete, firstkey, nextkey: data base subroutines delete on RC DOS dielette	dbm(3X) dosread(1)
dosread: read, write, dir, crypt, setkey, encrypt:	delete on PC-DOS diskette	crypt(3)
getdiskbyname: get disk	description by its name.	getdisk(3)
getsstype, setsent, endseent: get file system	descriptor file entry. /getfsspec, getfsfile,	getfsent(3)
access:	determine accessibility of a file.	access(3F)
tb: line discipline for digitizing	devices.	tb(4)
fimin, fimax, ffrac, dfimin, dfimax,	dffrac, inmax: return extreme values	flmin(3F)
flmin, flmax, ffrac, dflmin,	dflmax, dffrac, inmax: return extreme values	flmin(3F)
values. fimin, fimax, ffrac,	dflmin, dflmax, dffrac, inmax: return extreme	flmin(3F)
autoconf:	diagnostics from the autoconfiguration code	autoconf(4)
xclock - X Window System, analog /	digital clock.	xclock(1)
tb: line discipline for	digitizing devices	tb(4)
dosread: read, write,	dir, delete on PC-DOS diskette	dosread(1)
fpa:		fpa(3X)
	direct interface to floating point accelerator	ipa(3A)
cd: change	direct interface to floating point accelerator directory	csh(1)
chdir: change		csh(1) csh(1)
chdir: change chdir: change default	directory	csh(1) csh(1) chdir(3F)
chdir: change chdir: change default getcwd: get pathname of current working	directory	csh(1) csh(1) chdir(3F) getcwd(3F)
chdir: change chdir: change default getcwd: get pathname of current working scandir, alphasort: scan a	directory. directory. directory. directory. directory.	csh(1) csh(1) chdir(3F) getcwd(3F) scandir(3)
chdir: change chdir: change default getcwd: get pathname of current working scandir, alphasort: scan a unlink: remove a	directory. directory. directory. directory. directory. directory. directory.	csh(1) csh(1) chdir(3F) getcwd(3F) scandir(3) unlink(3F)
chdir: change chdir: change default getcwd: get pathname of current working scandir, alphasort: scan a unlink: remove a readdir, telldir, seekdir, rewinddir, closedir:	directory. directory. directory. directory. directory. directory. directory. directory entry. directory operations. opendir,	csh(1) csh(1) chdir(3F) getcwd(3F) scandir(3) unlink(3F) directory(3)
chdir: change chdir: change default getcwd: get pathname of current working scandir, alphasort: scan a unlink: remove a readdir, telldir, seekdir, rewinddir, closedir: getwd: get current working	directory. directory. directory. directory. directory. directory. directory. directory entry. directory operations. opendir, directory pathname.	csh(1) csh(1) chdir(3F) getcwd(3F) scandir(3) unlink(3F) directory(3) getwd(3)
chdir: change chdir: change default getcwd: get pathname of current working scandir, alphasort: scan a unlink: remove a readdir, telldir, seekdir, rewinddir, closedir:	directory. directory. directory. directory. directory. directory. directory. directory entry. directory operations. opendir,	csh(1) csh(1) chdir(3F) getcwd(3F) scandir(3) unlink(3F) directory(3)

	·	
unhash:	discard command hash table	csh(1)
unset:	discard shell variables	csh(1)
ap: asynchronous data mode protocol line	discipline	ap(4)
tb: line	discipline for digitizing devices	tb(4)
disk: format of reserved areas of the hard	disk	disk(4)
getdiskbyname: get	disk description by its name.	getdisk(3)
geteloko jilanie. get	disk: format of reserved areas of the hard disk	disk(4)
hd: hard		hd(4)
		` '
rvdcopy: copy contents of one RVD	disk pack to another	rvdcopy(8)
format: how to format	disk packs.	format(8V)
diskpart: calculate default	disk partition sizes	diskpart(8)
rvd: Remote Virtual	Disk protocol.	rvd(4p)
newvd: create a new filesystem on a Remote Virtual	Disk (RVD)	newvd(8)
vdspind: spin up or spin down a Remote Virtual	Disk (RVD). vdspin,	vdspin(2)
vddb: Remote Virtual	Disk (RVD) data base manager	vddb(8)
rvddown: force spindown of a Remote Virtual	Disk (RVD) pack	rvddown(8)
spinup, spindown: spin up/down Remote Virtual		
		spinup(8)
rvdexch: exchange names of two Remote Virtual	Disk (RVD) packs.	rvdexch(8)
rvdflush: spindown client's Remote Virtual	Disk (RVD) packs	rvdflush(8)
savephys: back up and restore Remote Virtual	Disk (RVD) packs to and from tape. /zaprvd,	savervd(8)
rvdchlog: change logging level of Remote Virtual	Disk (RVD) server	rvdchlog(8)
rvdgetm: get operations message from Remote Virtual	Disk (RVD) server	rvdgetm(8)
rvdsend - send control stream to Remote Virtual	Disk (RVD) server	rvdsend(8)
rvdsetm: set operations message on Remote Virtual	Disk (RVD) server	rvdsetm(8)
rvdshow: show connections to Remote Virtual	Disk (RVD) server	rvdshow(8)
rydshut: force shutdown of Remote Virtual	Disk (RVD) server.	rvdshut(8)
rvddb: Remote Virtual		
	Disk (RVD) server configuration table	rvddb(5)
rvdsrv: Remote Virtual	Disk (RVD) server daemon.	rvdsrv(8)
rvdlog: cause Remote Virtual	Disk (RVD) server to log statistics	rvdlog(8)
vdstats: acquire client Remote Virtual	Disk (RVD) statistics	vdstats(2)
vdstats: list client Remote Virtual	Disk (RVD) statistics	vdstats(8)
up, down: client Remote Virtual	Disk (RVD) utilities	up(1)
scsiformat: format the IBM 9332	disk unit.	scsiformat(8c)
dosread: read, write, dir, delete on PC-DOS	diskette	dosread(1)
fd:	diskette interface	fd(4)
fdformat: format	diskettes.	fdformat(8r)
		_ , ,
flcopy: copier for		flcopy(8r)
	diskpart: calculate default disk partition sizes	diskpart(8)
format: format hard	disks	format(8r)
rxformat: format floppy	disks	rxformat(8V)
information about client Remote Virtual	Disks (RVDs). /etc/rvd/rvdtab:	rvdtab(5)
(SCSI) Adapter. sc: IBM 9332	disks using the IBM Small Computer System Interface	sc(4)
error: analyze and	disperse compiler error messages	error(1)
the IBM Academic Information Systems experimental	display. aedemul: graphics interfaces for	aedemul(4)
xload - X window system load average	display.	xload(1)
ibm8604. ibm8514: IBM 8514/A	Display adapter for the ibm8503, ibm8513, ibm8514,	ibm8514(4)
the contract of the contract o		
bitprt: capture the image on a bitmap	display and print it on an IBM printer	bitprt(1)
aedjournal:	display commands in a log file.	aedjournal(1)
VI_EnCur, VI_DisCur, VI_PosnCur: control the	display cursor. VI_MDefnCur, VI_FDefnCur,	cursor(3G)
VI_MRead, VI_FRead: read	display data.	read(3G)
umask: change or	display file creation mask	csh(1)
intro: introduction to	display graphics subroutines	intro(3G)
cons: keyboard and console	display interface	cons(4)
ibm5081, mpel – IBM 5081 Mega Pel	Display interface.	ibm5081(4)
ibm5151, mono: IBM 5151 Monochrome	Display interface.	ibm5151(4)
	Display interface	ibm5154(4)
ibm5154, ega: IBM 5154 Enhanced Graphics		
apa8: IBM 6153 Advanced Monochrome Graphics	Display interface. ibm6153,	ibm6153(4)
ibm6154, apa8c: IBM 6154 Advanced Color Graphics	Display interface.	ibm6154(4)
apa16: IBM 6155 Extended Monochrome Graphics	Display interface. ibm6155,	ibm6155(4)
aed: IBM Academic Information Systems experimental	display interface. ibmaed,	ibmaed(4)
dumpaed: dump aed	display memory as a binary file	dumpaed(1)
dumpapal6; dump apal6	display memory as a binary file	dumpapa16(1)
dumpapa8: dump apa8	display memory as a binary file	dumpapa8(1)
dumpapa8c: dump apa8c	display memory as a binary file	dumpapa8c(1)
setscreen: control	display screen access	setscreen(8)
consoles: utility database of	display screens.	consoles(5)
IBM Academic Information Systems experimental	display self-tests. aedtest:	aedtest(8)
xfd - X window system font		
	dienlauer	xfd(1)
xlsfonts - X window system font list	displayer	
xprop - X Window System property	displayer	xlsfonts(1)
	displayer	xlsfonts(1) xprop(1)
hypot, cabs: Euclidean	displayer	xlsfonts(1)
res_mkquery, res_send, res_init,	displayer	xlsfonts(1) xprop(1)
	displayer	xlsfonts(1) xprop(1) hypot(3M)
res_mkquery, res_send, res_init,	displayer	xlsfonts(1) xprop(1) hypot(3M) resolver(3) resolver(3)
res_mkquery, res_send, res_init, res_mkquery, res_send, res_init, dn_comp, diskette.	displayer. displayer. distance, complex absolute value. dn_comp, dn_expand: resolver routines. dn_expand: resolver routines. dosread: read, write, dir, delete on PC-DOS	xlsfonts(1) xprop(1) hypot(3M) resolver(3) resolver(3) dosread(1)
res_mkquery, res_send, res_init, res_mkquery, res_send, res_init, dn_comp, diskette. vdabort: abort and spin	displayer	xlsfonts(1) xprop(1) hypot(3M) resolver(3) resolver(3)

up,	down: client Remote Virtual Disk (RVD) utilities	up(1)
rand,	drand, irand: return random values	rand(3F)
random.	drandm, irandm: better random number generator	random(3F)
VI Circle:	draw a circle.	circle(3G)
graph:	draw a graph.	graph(1G)
VI ALine, VI RLine:	draw a line.	line(3G)
VI_ABING, VI_KBING: VI String:	draw a string.	string(3G)
VI_MImage, VI_FImage:	draw an image.	image(3G)
pic: troff preprocessor for	drawing simple pictures.	pic(1)
exponent manipulations. copysign,	drem, finite, logb, scalb: copysign, remainder,	ieee(3M)
isnan, unordered, finite, infinity,/ copysign,	drem, logb, scalb, rint, classdouble, classfloat,	ieee(3)
vdabort: abort and spin down a		vdabort(8)
etime.	drive	· · · · · · · · · · · · · · · · · · ·
	•	etime(3F)
xpr: print X window	dump.	xpr(1)
dumpaed:	dump aed display memory as a binary file	dumpaed(1)
dumpapa16:	dump apa16 display memory as a binary file	
dumpapa8:	dump apa8 display memory as a binary file	dumpapa8(1)
dumpapa8c:	dump apa8c display memory as a binary file	dumpapa8c(1)
nfabort:	dump core and log it in a notesfile	nfabort(3)
landump:	dump IBM Token-Ring Personal Computer Adapter.	landump(8r)
C1	dumpaed: dump aed display memory as a binary file	dumpaed(1)
file.	dumpapa16: dump apa16 display memory as a binary .	dumpapa16(1)
file.	dumpapa8: dump apa8 display memory as a binary	dumpapa8(1)
file.	dumpapa8c: dump apa8c display memory as a binary	dumpapa8c(1)
xwd - X Window System, window image	dumper	xwd(1)
echo:	echo arguments	csh(i)
	echo: echo arguments	csh(1)
	ecvt, fcvt, gcvt: output conversion	ecvt(3)
end, etext,	edata: last locations in program	end(3)
ld: link	editor	ld(1)
bitmap: bitmap	editor for X window system.	bitmap(1)
a.out: assembler and link	editor output.	a.out(5)
ibm5154,	ega: IBM 5154 Enhanced Graphics Display interface.	ibm5154(4)
etime, dtime: return	elapsed execution time	etime(3F)
•		
insque, remque: insert/remove	element from a queue.	insque(3)
1.6.11.11.00	else: alternative commands	csh(1)
bufemul: kernel buffering	emulator	bufemul(4)
getfpemulator: return address of the floating-point	emulator	getfpemulator(2)
ibmemul: IBM 3101	emulator	ibmemul(4)
kbdemul: default keyboard	emulator	kbdemul(4)
stdemul: standard output	emulator	stdemul(4)
xterm: X window system terminal	emulator	xterm(1)
xemul: X input	emulator for queuing keyboard and mouse events	xemul(4)
Xtty: routines to provide terminal	emulator windows	xtty(3X)
crypt, setkey,	encrypt: DES encryption	crypt(3)
crypt, setkey, encrypt: DES	encryption	crypt(3)
	end, etext, edata: last locations in program	end(3)
logout:	end session.	
	end: terminate loop.	csh(1)
/getfsspec, getfsfile, getfstype, setfsent,		2 '
getgrent, getgrgid, getgrnam, setgrent,	endgrent: get group file entry.	getgrent(3)
gethostbyaddr, gethostent, sethostent,	endhostent: get network host entry. gethostbyname,	gethostbyname(3N)
G, B, b	endif: terminate conditional	csh(1)
getnetent, getnetbyaddr, getnetbyname, setnetent,	endnetcnt: get network entry.	getnetent(3N)
getprotobynumber, getprotobyname, setprotoent,	endprotoent: get protocol entry. getprotoent,	
getprotodynamic, getprotodynamic, setprotoent, getpwent, getpwuid, getpwnam, setpwent,	endpwent, setpwfile: get password file entry	getprotoch(314)
getservbyport, getservbyname, setservent,	endservent; get service entry. getservent,	getservent(3N)
getsel vojport, getsel vojname, setsel vent,	endsw: terminate switch	csh(1)
mattt		
getttyent, getttynam, setttyent,	endityent: get ttys file entry	getttyent(3)
getusershell, setusershell,	endusershell: get legal user shells.	getusershell(3)
ibm5154, ega: IBM 5154	Enhanced Graphics Display interface	ibm5154(4)
nlist: get	entries from name list.	nlist(3)
setssent, endssent: get file system descriptor file	entry. getssent, getssspec, getssfile, getsstype,	getfsent(3)
getgrnam, setgrent, endgrent: get group file	entry. getgrent, getgrgid,	getgrent(3)
sethostent, endhostent: get network host	entry. gethostbyname, gethostbyaddr, gethostent,	gethostbyname(3N)
getnetbyname, setnetent, endnetent: get network	entry. getnetent, getnetbyaddr,	getnetent(3N)
setprotoent, endprotoent: get protocol	entry. /getprotobynumber, getprotobyname,	getprotoent(3N)
setpwent, endpwent, setpwfile: get password file	entry. getpwent, getpwuid, getpwnam,	getpwent(3)
getservbyname, setservent, endservent: get service	entry. getservent, getservbyport,	getser vent(3N)
getttynam, setttyent, endttyent: get ttys file	entry. getttyent,	getttyent(3)
unlink: remove a directory	entry.	unlink(3F)
execv, execle, execlp, execvp, exec, execve, exect,	environ: execute a file. exect,	execi(3)
seteny: set variable in	environment.	csh(1)
getenv: value for	environment name.	getenv(3)
unsetenv: remove	environment variables	csh(1)
getenv: get value of		getenv(3F)
gotoniii. got value oi	**************************************	P/

linemod, space, closepl: graphics/ plot: openpl,	erase, label, line, circle, arc, move, cont, point,	plot(3X)
	erf, erfc: error functions.	` '
erf,	erfc: error functions.	
messages.	error: analyze and disperse compiler error	
erf, erfc:	error functions.	
error: analyze and disperse compiler	error messages.	` ,
perror, sys_errlist, sys_nerr: system	error messages.	perror(3)
perror, gerror, ierrno: get system	error messages.	perror(3F)
intro: introduction to system calls and	error numbers.	* .
traper: trap arithmetic	errors.	
Virtual Disks (RVDs).	/etc/rvd/rvdtab: information about client Remote	rvdtab(5)
end,	etext, edata: last locations in program.	
un: IBM RT PC Baseband Adapter for use with	Ethernet	
• • •	etime, dtime: return elapsed execution time.	
hypot, cabs:	Euclidean distance, complex absolute value	71 (,
***	eval: re-evaluate shell data	
history: print history	event list.	· •
X input emulator for queuing keyboard and mouse	events. xemul:	
packs. rvdexch:	exchange names of two Remote Virtual Disk (RVD)	rvdexch(8)
execl, execv, execle, execlp, execvp,	exec, execve, exect, environ: execute a file	
	exec: overlay shell with specified command	
exect, environ: execute a file.	execl, execv, execle, execlp, execvp, exec, execve,	
environ: execute a file. execl, execv,	execle, execlp, execvp, exec, execve, exect,	
execute a file. execl, execv, execle, execvl, execve, execle, execvp, execve, execve,	execlp, execvp, exec, execve, exect, environ:	
	exect, environ: execute a file.	
execlp, execvp, exec, execve, exect, environ:	execute a file. exect, execv, execte,	
alarm:	execute a subroutine after a specified time	
system:	execute a UNIX command	
repeat: aedrunner:	execute command repeatedly	
sleep: suspend	execution for an interval.	
sleep: suspend	execution for interval.	
usleep: suspend	execution for interval.	
monitor, monstartup, moncontrol: prepare etime, dtime: return elapsed	execution profile.	` '
environ: execute a file. exect,		• • •
exect, execv, execte, exectp, execvp, exec,	execv, execle, execlp, execvp, exec, execve, exect, execve, exect, environ: execute a file	execl(3)
file. execl, execve, execve, execve, execle, execle,		
link: make a link to an	execvp, exec, execve, exect, environ: execute a existing file	` '
breaksw:	exit from switch.	
Oleaksw.		
pending output.	exit: leave shell	
pending output.	exit: terminate a process after husbing any	
break:	exit while/foreach loop.	` '
logarithm, power.	exp, expm1, log, log10, log1p, pow: exponential,	
glob: filename	expand argument list.	
interfaces for the IBM Academic Information Systems	experimental display. aedemul: graphics	
ibmaed, aed: IBM Academic Information Systems	experimental display interface	
aedtest: IBM Academic Information Systems	experimental display self-tests.	
logarithm, power. exp.	expm1, log, log10, log1p, pow: exponential,	
frexp, ldexp, modf: split into mantissa and	exponent	frexp(3)
drem, finite, logb, scalb: copysign, remainder,	exponent manipulations. copysign,	
exp, expm1, log, log10, log1p, pow:	exponential, logarithm, power	
re comp, re exec: regular	expression handler	* *
ibm6155, apa16: IBM 6155	Extended Monochrome Graphics Display interface	ibm6155(4)
	f77: FORTRAN 77 compiler	
ioinit: change	f77 I/O initialization	
libraries plot: openpl et al.:	f77 library interface to plot (3X)	plot(3F)
tclose, tread, twrite, trewin, tskipf, tstate:	f77 tape I/O. topen,	
functions.	fabs, floor, ceil: absolute value, floor, ceiling	floor(3M)
sigvec: software signal	facilities	sigvec(2)
signal: simplified software signal	facilities	signal(3C)
malloc, free,	falloc: memory allocator	malloc(3F)
abort: generate a	fault	abort(3)
trpfpe, fpecnt: trap and repair floating point	faults	trpfpe(3F)
chmod,	fchmod: change mode of file.	chmod(2)
chown,	fchown: change owner and group of a file	chown(2)
	fclose, fflush: close or flush a stream	fclose(3S)
ecvt,	fcvt, gcvt: output conversion	
	fd: diskette interface.	fd(4)
	fdate: return date and time in an ASCII string	
	fdformat: format diskettes	fdformat(8r)
utility.	fdisk: boot record partition table maintenance	
fopen, freopen,	fdopen: open a stream	
ferror,	feof, clearerr, fileno: stream status inquiries	
inquiries	ferror feof clearery fileno; stream status	ferror(3S)

subroutines. dbminit,	fetch, store, delete, firstkey, nextkey: data base	dbm(3X)
fclose,	fflush: close or flush a stream	fclose(3S)
extreme values. fimin, fimax,	ffrac, dflmin, dflmax, dffrac, inmax: return	flmin(3F)
bcopy, bcmp, bzero,	ffs: bit and byte string operations	bstring(3)
· · · · · ·	fg: bring job into foreground.	csh(1)
getc, getc, getchar,	fgetc: get a character from a logical unit fgetc, getw: get character or word from stream	getc(3F) getc(3S)
gett, getthar,	fgets: get a string from a stream.	gets(3S)
aedjournal: display commands in a log	file.	aedjournal(1)
aedrunner: execute graphics commands in a log	file	aedrunner(1)
chmod, fchmod: change mode of	file.	chmod(2)
chown, fchown: change owner and group of a	file	chown(2)
core: format of memory image	file	core(5)
source: read commands from	file.	csh(1)
dumpaed: dump aed display memory as a binary	file	dumpaed(1)
dumpapa16: dump apa16 display memory as a binary	file	dumpapa16(1)
dumpapa8: dump apa8 display memory as a binary	file.	dumpapa8(1) dumpapa8c(1)
dumpapa8c: dump apa8c display memory as a binary access: determine accessibility of a	file.	access(3F)
chmod: change mode of a	file.	chmod(3F)
execvp, exec, execve, exect, environ: execute a	file. exect, exect, execte, exectp,	exect(3)
link: make a link to an existing	file.	link(3F)
begin logging subroutine calls and close a log	file. VI_Login, VI_Logout:	log(3G)
rename: rename a	file.	rename(3F)
VI Run: process a log	file	run(3G)
umask: change or display	file creation mask	csh(1)
setfsent, endfsent: get file system descriptor	file entry. /getssspec, getssfile, getsstype,	getfsent(3)
getgrgid, getgrnam, setgrent, endgrent: get group	file entry. getgrent,	getgrent(3)
setpwent, endpwent, setpwfile: get password	file entry. getpwent, getpwuid, getpwnam,	getpwent(3)
getttynam, setttyent, endttyent: get ttys	file entry. getttyent,	getttyent(3)
rc.config: configuration	file for startup scripts.	rc.config(5)
mktemp: make a unique	file name.	mktemp(3)
fseek, ftell: reposition a stat, lstat, fstat: get	file on a logical unit	fseek(3F) stat(3F)
newfs: construct a new	file system.	newfs(8)
getfsfile, getfstype, setfsent, endfsent: get	file system descriptor file entry. /getfsspec,	getfsent(3)
restore: incremental	file system restore	restore(8)
utime: set	file times.	utime(3C)
truncate, ftruncate: truncate a	file to a specified length	truncate(2)
glob:	filename expand argument list	csh(1)
ferror, feof, clearerr,	fileno: stream status inquiries	ferror(3S)
config: build system configuration	files	config(8)
makedev: make system special	files.	makedev(8)
omerge: merge object	files.	omerge(8)
intro: introduction to special pprint: print text	files and hardware support	intro(4) pprint(1)
ptroff: print text	files on IBM 3812 Pageprinter	ptroff(1)
badsect: create	files to contain bad sectors.	badsect(8)
newvd: create a new	filesystem on a Remote Virtual Disk (RVD)	newvd(8)
5152 Graphics Printer nroff post-processing	filter. prfl: IBM 4201 Proprinter/IBM	prfl(1)
colpro: column	filter for IBM 4201 Proprinter	colpro(1)
ppt: spooling system	filter for the IBM 3812 Pageprinter	ppt(8)
plot: graphics	filters	plot(1G)
Graphics Printer. ibmbit, ibmgra, ibmpro: output	filters for the IBM 4201 Proprinter and IBM 5152	lpfilter(8r)
ttyname, isatty, ttyslot:	find name of a terminal.	ttyname(3)
ttynam, isatty:	find name of a terminal port.	ttynam(3F)
rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem,	finite, infinity, nextdouble, nextfloat, / /scalb, finite, logb, scalb: copysign, remainder, exponent	ieee(3) ieee(3M)
plot: openpl et al.: f77 library interface to	plot (3X) libraries	plot(3F)
dbminit, fetch, store, delete,	firstkey, nextkey: data base subroutines.	dbm(3X)
ptfinstall: install a Program Temporary	Fix (PTF)	ptfinstall(8)
Handler.	xmh: X window interface to the mh Mail	xmh(1)
arff,	floopy: archiver and copier for floppy	arff(8V)
	flcopy: copier for diskettes	flcopy(8r)
extreme values. flmin,	fimax, ffrac, dfimin, dfimax, dffrac, inmax: return	flmin(3F)
return extreme values.	fimin, fimax, ffrac, dfimin, dfimax, dffrac, inmax:	flmin(3F)
afpacode: load, test, and bring online the Advanced	Floating Point Accelerator	afpacode(8r)
fpa: direct interface to	floating point accelerator.	fpa(3X)
trpfpe, fpecnt: trap and repair	floating point faults	trpfpe(3F)
trapov: trap and repair getfloatstate: return machine and process	floating point overflow	trapov(3F)
gethoatstate: return machine and process getfpemulator: return address of the	floating point state	getfloatstate(2) getfpemulator(2
infran: signals invalid	floating-point operations on a VAX (temporary).	infnan(3M)
functions. fabs,	floor, ceil: absolute value, floor, ceiling	floor(3M)
fabs, floor, ceil: absolute value,	floor, ceiling functions.	floor(3M)
arff floory archiver and copier for		arff(RV)

		6 .464.0
rxformat: format	• • •	rxformat(8V)
fclose, fflush: close or	flush a stream. flush: flush output to a logical unit.	fclose(3S) flush(3F)
flush:	flush output to a logical unit.	flush(3F)
exit: terminate a process after	flushing any pending output.	exit(3)
/gcd, invert, rpow, msqrt, mcmp, move, min, omin,	fmin, m_in, mout, omout, fmout, m_out, sdiv, itom:/	mp(3X)
/mcmp, move, min, omin, fmin, m_in, mout, omout,	fmout, m_out, sdiv, itom: multiple precision/	mp(3X)
xfd - X window system	font displayer	xfd(1)
xlsfonts - X window system	font list displayer.	xlsfonts(1)
font3812:	font structures for 3812 fonts font3812: font structures for 3812 fonts	font3812(5) font3812(5)
font3812: font structures for 3812	fonts.	font3812(5)
VI_GetFont, VI_DropFont: select and manipulate	fonts. VI Font.	font(3G)
build width tables for IBM 3812 Pageprinter	fonts. width3812:	width3812(8)
/cvt20to12, cvt00to12: convert IBM 3820 and IBM 3800	fonts for use with the IBM 3812 Pageprinter	cvt3812(8)
	fopen, freopen, fdopen: open a stream	fopen(3S)
VI Force:	force output of graphics orders	force(3G)
rvddown:	force shutdown of Remote Virtual Disk (RVD) server force spindown of a Remote Virtual Disk (RVD) pack	rvdshut(8) rvddown(8)
i vado wii.	foreach: loop over list of names	csh(1)
fg: bring job into	foreground	csh(1)
	fork: create a copy of this process	fork(3F)
idate, itime: return date or time in numerical		idate(3F)
format: how to	format disk packs.	format(8V)
fdformat:	format diskettes	fdformat(8r) rxformat(8V)
rxformat:	format floppy disks.	format(8r)
format:	format hard disks.	format(8r)
	format: how to format disk packs	format(8V)
core:	format of memory image file	core(5)
disk:	format of reserved areas of the hard disk	disk(4)
scsiformat:	format the IBM 9332 disk unit.	scsiformat(8c)
scanf, fscanf, sscanf: printf, fprintf, sprintf:	formatted input conversion	scanf(3S) printf(3S)
printi, iprinti, sprinti. f77:	FORTRAN 77 compiler	f77(1)
intro: introduction to	FORTRAN library functions.	intro(3F)
putc, fputc: write a character to a	fortran logical unit.	putc(3F)
accelerator.	fpa: direct interface to floating point	fpa(3X)
trpfpe,	fpecnt: trap and repair floating point faults	trpfpe(3F)
printf,	fprintf, sprintf: formatted output conversion.	printf(3S)
fptestround, fpsetround, swapround, fptestflag, /infinity, nextdouble, nextfloat, fptestround,	fpsetflag, /infinity, nextdouble, nextfloat, fpsetround, swapround, fptestflag, fpsetflag,	ieee(3)
nextfloat, fptestround, fpsetround, swapround,	fptestflag, fpsetflag, /infinity, nextdouble,	ieee(3)
/unordered, finite, infinity, nextdouble, nextfloat,	fptestround, fpsetround, swapround, fptestflag,/	ieee(3)
putc, putchar,	fputc, putw: put character or word on a stream	putc(3S)
putc,	fputc: write a character to a fortran logical unit	putc(3F)
puts,		puts(3S)
allaa	fread, fwrite: buffered binary input/output	fread(3S)
mailoc,	free, falloc: memory allocator	malloc(3F) malloc(3)
fopen,	freopen, fdopen: open a stream.	fopen(3S)
exponent.	frexp, ldexp, modf: split into mantissa and	frexp(3)
scanf,	fscanf, sscanf: formatted input conversion	scanf(3S)
	fseek, ftell: reposition a file on a logical unit.	fseek(3F)
mant 1-a-a	fseek, ftell, rewind: reposition a stream	fseek(3S)
stat, Istat, fseek,	fstat: get file status	stat(3F) fseek(3F)
fseek,	ftell, rewind: reposition a stream.	fseek(3S)
time,	ftime: get date and time.	time(3C)
truncate,	ftruncate: truncate a file to a specified length	truncate(2)
tn3270:	full-screen remote login to IBM VM/CMS	tn3270(1)
lgamma: log gamma	function.	Igamma(3M)
asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise	functions	asinh(3M) bit(3F)
erf, erfc: error	functions.	erf(3M)
fabs, floor, ceil: absolute value, floor, ceiling	functions	floor(3M)
intro: introduction to C library	functions	intro(3)
intro: introduction to FORTRAN library	functions	intro(3F)
j0, j1, jn, y0, y1, yn: Bessel	functions	j0(3M)
math: introduction to mathematical library sinh, cosh, tanh: hyperbolic	functions	math(3M) sinh(3M)
cos, tan, asin, acos, atan, atan2: trigonometric	functions and their inverses. sin,	sin(3M)
bessel	functions: of two kinds for integer orders.	bessel(3F)
curses: screen	functions with "optimal" cursor motion.	curses(3X)
fread,	fwrite: buffered binary input/output	fread(3S)
lgamma: log	gamma function.	Igamma(3M)

fmin, m_in, mout,/ madd, msub, mult, mdiv, pow,		mp(3X)
ecvt, fcvt,		ecvt(3)
abort: rand, srand: random number	generate a fault	
random, drandm, irandm: better random number	generator	
/srandom, initstate, setstate: better random number	generator; routines for changing generators	random(3)
random number generator; routines for changing	generators. /srandom, initstate, setstate: better	
perror,		
	getarg, large: return command line arguments	
C	getc, fgetc: get a character from a logical unit	
from stream.	getc, getchar, fgetc, getw: get character or word	
stream. getc,	getchar, fgetc, getw: get character or word from getcwd: get pathname of current working directory	
	getdiskbyname: get disk description by its name	
	getenv: get value of environment variables	
	getenv: value for environment name	
point state.	getfloatstate: return machine and process floating	
emulator.	getspemulator: return address of the floating-point	
setfsent, endfsent: get file system descriptor/	getseent, getsspec, getssfile, getsstype,	
system descriptor file entry. getfsent, getfsspec, endfsent: get file system descriptor/ getfsent,	getfsfile, getfstype, setfsent, endfsent: get file getfsspec, getfsfile, getfstype, setfsent,	
descriptor file/ getfsent, getfsspec, getfsfile,	getfstype, setfsent, endfsent: get file system	
getuid,	getgid: get user or group ID of the caller	getuid(3F)
get group file entry.	getgrent, getgrgid, getgrnam, setgrent, endgrent:	
file entry. getgrent,	getgrgid, getgrnam, setgrent, endgrent: get group	
getgrent, getgrgid,	getgrnam, setgrent, endgrent: get group file entry	
get network host entry. gethostbyname,	gethostbyaddr, gethostent, sethostent, endhostent:	
sethostent, endhostent: get network host entry.	gethostbyname, gethostbyaddr, gethostent,	
host entry. gethostbyname, gethostbyaddr,	gethostent, sethostent, endhostent: get network getlog: get user's login name	
	getlogin: get login name.	
get network entry. getnetent,	getnetbyaddr, getnetbyname, setnetent, endnetent:	
entry. getnetent, getnetbyaddr,		getnetent(3N)
endnetent: get network entry.	getnetent, getnetbyaddr, getnetbyname, setnetent,	
•	getopt: get option letter from argv	
	getpass: read a password	
	getpid: get process id.	
protocol entry. getprotoent, getprotobynumber,	getprotobyname, setprotoent, endprotoent: get	
endprotoent: get protocol entry. getprotoent, setprotoent, endprotoent: get protocol entry.	getprotobynumber, getprotobyname, setprotoent, getprotobynumber, getprotobyname,	
scrprotocne, enaprotocne, get protocoi enti y.	getpw: get name from uid.	
setpwfile: get password file entry.	getpwent, getpwuid, getpwnam, setpwent, endpwent,	
password file entry. getpwent, getpwuid,	getpwnam, setpwent, endpwent, setpwfile: get	
get password file entry. getpwent,	getpwuid, getpwnam, setpwent, endpwent, setpwfile: .	getpwent(3)
	gets, fgets: get a string from a stream.	
entry. getservent, getservbyport,	getservbyname, setservent, endservent: get service	
endservent: get service entry. getservent, setservent, endservent: get service entry.	getservbyport, getservbyname, setservent, getservbyport, getservbyname,	getservent(3N)
ttys file entry.	gettervent, gettervoyport, getservoyname, getttyent, getttynam, setttyent, endttyent: get	
entry. getttyent,	getttynam, setttyent, endttyent; get ttys file	
, Berryen,	getuid, getgid: get user or group ID of the caller	getuid(3F)
user shells.	getusershell, setusershell, endusershell: get legal	
getc, getchar, fgetc,	getw: get character or word from stream	
	getwd: get current working directory pathname	
ASCII atima lasaltima	glob: filename expand argument list	
ASCII. ctime, localtime, time, ctime, ltime,	gmtime, asctime, timezone: convert date and time to gmtime: return system time	ctime(3) time(3F)
setjmp, longjmp: non-local	goto	setjmp(3)
200Jp., 101p., 1101	goto: command transfer.	csh(1)
graph: draw a	graph	graph(1G)
	graph: draw a graph	graph(1G)
aedrunner: execute	graphics commands in a log file	aedrunner(1)
ibm5154, ega: IBM 5154 Enhanced ibm6153, apa8: IBM 6153 Advanced Monochrome	Graphics Display interface	ibm5154(4) ibm6153(4)
ibm6154, apa8c: IBM 6154 Advanced Color	Graphics Display interface	ibm6154(4)
ibm6155, apa16: IBM 6155 Extended Monochrome	Graphics Display interface	ibm6155(4)
plot:	graphics filters.	plot(1G)
arc, move, cont, point, linemod, space, closepl:	graphics interface. /erase, label, line, circle,	plot(3X)
Information Systems experimental display. aedemul:	graphics interfaces for the IBM Academic	aedemul(4)
VI_Force: force output of	graphics orders.	force(3G)
VI_QFont, VI_QMerge, VI_QPoint, VI_QWidth: query	graphics parameters. /VI_QColor, VI_QDash,	query(3G)
filters for the IBM 4201 Proprinter and IBM 5152 nroff for the IBM 4201 Proprinter and IBM 5152	Graphics Printer, ibmbit, ibmgra, ibmpro: output	ipfilter(8r)
prfl: IBM 4201 Proprinter and IBM 5152	Graphics Printer. proff:	proff(1) prff(1)
intro: introduction to display	graphics subroutines	
lib 1649; subrouting for the UD 1649	graphics subjournes.	1:52649/2V)

n to the constitution	the state of the s	
Pageprinter. vgrind:	grind nice listings of programs for the IBM 3812	vgrind(1)
initgroups: initialize	group access list.	initgroups(3)
getgrgid, getgrnam, setgrent, endgrent: get setruid, setgid, setegid, setrgid: set user and	group file entry. getgrent,	getgrent(3) setuid(3)
getuid, getgid: get user or chown, fchown: change owner and	group ID of the caller	getuid(3F)
	gtty: set and get terminal state (defunct).	chown(2)
stty, stop:	halt a job or process	stty(3C) csh(1)
stop.	halt: stop the processor.	halt(8)
re comp, re exec: regular expression	handler.	`
xmh: X window interface to the mh Mail	Handler	regex(3) xmh(1)
nohup: run command immune to	hangups.	csh(1)
crash: what	happens when the system crashes.	crash(8r)
crash: what	happens when the system crashes	crash(8V)
disk: format of reserved areas of the	hard disk.	disk(4)
hd:	hard disk interface.	hd(4)
format: format	hard disks.	format(8r)
support:	hardware and software support information	support(1)
intro: introduction to special files and	hardware support.	intro(4)
rehash: recompute command	hash table.	csh(1)
unhash: discard command	hash table.	csh(1)
hashstat: print command	hashing statistics	csh(1)
1	hashstat: print command hashing statistics	csh(1)
•	hc: High C compiler	hc(1)
	hd: hard disk interface	hd(4)
he:	High C compiler	hc(1)
history: print	history event list	csh(1)
	history: print history event list	csh(1)
hostnm: get name of current	host	hostnm(3F)
htonl, htons, ntohl, ntohs: convert values between	host and network byte order	byteorder(3N)
gethostent, sethostent, endhostent: get network	host entry. gethostbyname, gethostbyaddr,	gethostbyname(3N)
	hostnm: get name of current host	hostnm(3F)
format:	how to format disk packs	format(8V)
lib2648: subroutines for the	HP 2648 graphics terminal	lib2648(3X)
host and network byte order.	htonl, htons, ntohl, ntohs: convert values between	byteorder(3N)
and network byte order. htonl,	htons, ntohl, ntohs: convert values between host	byteorder(3N)
asinh, acosh, atanh: inverse	hyperbolic functions	asinh(3M)
sinh, cosh, tanh:	hyperbolic functions.	sinh(3M)
value.	hypot, cabs: Euclidean distance, complex absolute	hypot(3M)
getarg,	iargo: return command line arguments	getarg(3F)
ibmemul:	IBM 3101 emulator.	ibmemul(4)
mset: retrieve ASCII to	IBM 3270 keyboard map	mset(1)
data base for mapping ASCII keystrokes into	IBM 3270 keys. map3270:	map3270(5)
cvt3812, cvt20to12, cvt00to12: convert IBM 3820 and IBM 3820 and IBM 3800 fonts for use with the	IBM 3800 fonts for use with the IBM 3812/	cvt3812(8)
pprint: print text files on	IBM 3812 Pageprinter. /cvt00to12: convert IBM 3812 Pageprinter	cvt3812(8)
ppt: spooling system filter for the	IBM 3812 Pageprinter	pprint(1) ppt(8)
ptroff: print troff files on	IBM 3812 Pageprinter	
	IBM 3812 Pageprinter	vgrind(1)
width3812: build width tables for	IBM 3812 Pageprinter fonts.	width3812(8)
ibm3812pp:	IBM 3812 Pageprinter server	ibm3812pp(8)
printer3812:	IBM 3812 Pageprinter status information	printer3812(5)
3812/ cvt3812, cvt20to12, cvt00to12: convert	IBM 3820 and IBM 3800 fonts for use with the IBM .	cvt3812(8)
colpro: column filter for	IBM 4201 Proprinter	colpro(1)
ibmbit, ibmgra, ibmpro: output filters for the	IBM 4201 Proprinter and IBM 5152 Graphics Printer	lpfilter(8r)
proff: nroff for the	IBM 4201 Proprinter and IBM 5152 Graphics Printer	proff(1)
post-processing filter. prfl:	IBM 4201 Proprinter/IBM 5152 Graphics Printer nroff	prfl(1)
ibm5081, mpel –	IBM 5081 Mega Pel Display interface	ibm5081(4)
ibm5151, mono:	IBM 5151 Monochrome Display interface	ibm5151(4)
output filters for the IBM 4201 Proprinter and	IBM 5152 Graphics Printer. ibmbit, ibmgra, ibmpro: .	lpfilter(8r)
proff: nroff for the IBM 4201 Proprinter and	IBM 5152 Graphics Printer	proff(1)
ibm5154, ega:	IBM 5154 Enhanced Graphics Display interface	ibm5154(4)
interface. ibm6153, apa8:	1BM 6153 Advanced Monochrome Graphics Display .	ibm6153(4)
ibm6154, apa8c:	IBM 6154 Advanced Color Graphics Display interface	ibm6154(4)
interface. ibm6155, apa16:	IBM 6155 Extended Monochrome Graphics Display .	ibm6155(4)
ibm8513, ibm8514, ibm8604. ibm8514:	IBM 8514/A Display adapter for the ibm8503,	ibm8514(4)
scsiformat: format the	IBM 9332 disk unit.	scsiformat(8c)
Interface (SCSI) Adapter. sc:	IBM 9332 disks using the IBM Small Computer System	sc(4)
display. aedemul: graphics interfaces for the	IBM Academic Information Systems experimental	aedemul(4)
display interface. ibmaed, aed: display self-tests. aedtest:	IBM Academic Information Systems experimental	ibmaed(4)
the image on a bitmap display and print it on an	IBM Academic Information Systems experimental	aedtest(8)
debug: debugger for the	IBM printer. bitprt: capture	bitprt(1) debug(8)
kbdlock: lock the keyboard of the	IBM RT PC.	kbdlock(1)
un:		
		un(4)
lan:	IBM RT PC Baseband Adapter for use with Ethernet IBM RT PC Token-Ring Adapter	un(4) lan(4)

sc: IBM 9332 disks using the	IBM Small Computer System Interface (SCSI) Adapter.	sc(4)
landump: dump tn3270: full-screen remote login to	IBM Token-Ring Personal Computer Adapter IBM VM/CMS	
th3270. Iun-screen remote login to	ibm3812pp: IBM 3812 Pageprinter server	
interface.	ibm5081, mpel – IBM 5081 Mega Pel Display	
interface.	ibm5151, mono: IBM 5151 Monochrome Display	
interface.	ibm5154, ega: IBM 5154 Enhanced Graphics Display .	ibm5154(4)
Graphics Display interface.	ibm6153, apa8: IBM 6153 Advanced Monochrome	ibm6153(4)
Display interface.	ibm6154, apa8c: IBM 6154 Advanced Color Graphics .	ibm6154(4)
Graphics Display interface.	ibm6155, apa16: IBM 6155 Extended Monochrome	ibm6155(4)
ibm8514: IBM 8514/A Display adapter for the	ibm8503, ibm8513, ibm8514, ibm8604	• • •
	ibm8503, ibm8513, ibm8514, ibm8604	
IBM 8514/A Display adapter for the ibm8503,	ibm8513, ibm8514, ibm8604. ibm8514:	
ibm8503,	ibm8513, ibm8514, ibm8604	
ibm8503, ibm8513, ibm8514, ibm8604. 8514/A Display adapter for the ibm8503, ibm8513,	ibm8514: IBM 8514/A Display adapter for the ibm8514, ibm8604. ibm8514: IBM	
ibm8503, ibm8513,	ibm8514, ibm8604	
Display adapter for the ibm8503, ibm8513, ibm8514,	ibm8604. ibm8514: IBM 8514/A	
ibm8503, ibm8513, ibm8514,	ibm8604	
experimental display interface.	ibmaed, aed: IBM Academic Information Systems	
4201 Proprinter and IBM 5152 Graphics Printer.	ibmbit, ibmgra, ibmpro: output filters for the IBM	
· · · · · · · · · · · · · · · · · · ·	ibmemul: IBM 3101 emulator	ibmemul(4)
Proprinter and IBM 5152 Graphics Printer. ibmbit,	ibmgra, ibmpro: output filters for the IBM 4201	lpfilter(8r)
and IBM 5152 Graphics Printer. ibmbit, ibmgra,	ibmpro: output filters for the IBM 4201 Proprinter	lpfilter(8r)
getpid: get process	id	getpid(3F)
setgid, setegid, setrgid: set user and group	ID. setuid, seteuid,	setuid(3)
getuid, getgid: get user or group	ID of the caller	
form.	idate, itime: return date or time in numerical	idate(3F)
perror, gerror,	ierrno: get system error messages.	perror(3F)
	if: conditional statement.	csh(1)
	ifconfig: configure network interface parameters	ifconfig(8c)
scale: resize a bitmap abort: terminate abruptly with memory	image	scale(1) abort(3F)
VI_MImage, VI_FImage: draw an	image	image(3G)
xwd - X Window System, window	image dumper.	xwd(1)
core: format of memory	image file.	core(5)
printer. bitprt: capture the	image on a bitmap display and print it on an IBM	bitprt(1)
xwud - X Window System, window	image undumper.	xwud(l)
notify: request	immediate notification	csh(1)
nohup: run command	immune to hangups.	csh(1)
restore:	incremental file system restore	restore(8)
tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal	independent operation routines. tgetent,	termcap(3X)
objects.	index, rindex, Inblnk, len: tell about character	index(3F)
strncat, strcmp, strncmp, strcpy, strncpy, strlen, inet lnaof, inet netof: Internet address/	index, rindex: string operations. strcat,	string(3) inet(3N)
inet_addr, inet_network, inet_ntoa, inet_makeaddr,	inet Inaof, inet network, met makeaddr	
address/ inet addr, inet network, inet ntoa,	inet makeaddr, inet Inaof, inet netof: Internet	inet(3N)
/inet_network, inet_ntoa, inet_makeaddr, inet_lnaof,	inet netof: Internet address manipulation routines	
inet netof: Internet address/ inet addr,	inet network, inet ntoa, inet makeaddr, inet lnaof	inet(3N)
Internet address/ inet addr, inet network,	inet ntoa, inet makeaddr, inet lnaof, inet netof:	inet(3N)
/classdouble, classfloat, isnan, unordered, finite,	infinity, nextdouble, nextfloat, fptestround,/	ieee(3)
on a VAX (temporary).	infnan: signals invalid floating-point operations	infnan(3M)
dbx: dbx symbol table	information	dbx(5)
printer3812: IBM 3812 Pageprinter status	information	printer3812(5
support: hardware and software support	information.	support(1)
(RVDs). /etc/rvd/rvdtab:	information about client Remote Virtual Disks	rvdtab(5)
vtimes: get xwininfo - X Window System window	information about resource utilization information summarizer	vtimes(3C) xwininfo(1)
aedemul: graphics interfaces for the IBM Academic	Information Systems experimental display	aedemul(4)
ibmaed, aed: IBM Academic	Information Systems experimental display interface	ibmaed(4)
self-tests. aedtest: IBM Academic	Information Systems experimental display	aedtest(8)
	init: process control initialization.	init(8)
	initgroups: initialize group access list	initgroups(3)
init: process control	initialization.	init(8)
ioinit: change f77 I/O	initialization.	ioinit(3F)
VI_Init, VI_Term:	initialize and terminate the subroutine interface	init(3G)
initgroups:	initialize group access list	initgroups(3)
xinit - X window system	initializer	xinit(1)
popen, pclose:	initiate I/O to/from a process	popen(3)
generator; routines for changing/ random, srandom, fimin, fimax, ffrac, dfimin, dfimax, dffrac,	initstate, setstate: better random number inmax: return extreme values	random(3) fimin(3F)
scanf, fscanf, sscanf: formatted	input conversion.	scanf(3S)
events. xemul: X	input conversion	xemui(4)
ungetc: push character back into	input stream.	ungetc(3S)
fread, fwrite: buffered binary	input/output.	fread(3S)
stdio: standard buffered	input/output package	stdio(3S)

former foot eleganer flower streets status	tu audataa	for(25)
ferror, feof, clearerr, fileno: stream status insque, remque:	inquiries	
moque, remque.	insque, remque: insert/remove element from a queue	
ptfinstall:	install a Program Temporary Fix (PTF)	* '
restore.tape, restore.net:	install system from tape or over network	
learn: computer aided	instruction about UNIX	learn(1)
asy: multi-port asynchronous communications RS232C cons: keyboard and console display	interface	asy(4) cons(4)
fd: diskette	interface	fd(4)
hd: hard disk	interface.	hd(4)
ibm5081, mpel – IBM 5081 Mega Pel Display	interface	ibm5081(4)
ibm5151, mono: IBM 5151 Monochrome Display	interface	ibm5151(4)
ibm5154, ega: IBM 5154 Enhanced Graphics Display apa8: IBM 6153 Advanced Monochrome Graphics Display	interface	ibm5154(4) ibm6153(4)
apa8c: IBM 6154 Advanced Color Graphics Display	interface. ibm6153,	ibm6154(4)
IBM 6155 Extended Monochrome Graphics Display	interface. ibm6155, apa16:	ibm6155(4)
Academic Information Systems experimental display	interface. ibmaed, aed: IBM	ibmaed(4)
mouse: mouse	interface	mouse(4)
mtio: 4.3/RT magtape	interface	mtio(4)
psp: planar serial port RS232C speaker: console speaker	interface	psp(4) speaker(4)
st: streaming-tape	interface	st(4)
tty: general terminal	interface	tty(4)
VI_Term: initialize and terminate the subroutine	interface. VI_Init,	init(3G)
cont, point, linemod, space, closepl: graphics	interface. /erase, label, line, circle, arc, move,	plot(3X)
Xlib: C Language X Window System	Interface Library	Xlib(3X)
ifconfig: configure network syscall: system call	interface parameters	ifconfig(8c) syscall(8)
IBM 9332 disks using the IBM Small Computer System	Interface (SCSI) Adapter. sc:	sc(4)
plot: openpl et al.: f77 library	interface to plot (3X) libraries	plot(3F)
fpa: direct	interface to floating point accelerator	fpa(3X)
xmh: X window nfcomment: a user	interface to the mh Mail Handler	xmh(1) nfcomment(3)
experimental display. aedemul: graphics	interface to the notesfile system	aedemul(4)
/inet ntoa, inet makeaddr, inet lnaof, inet netof:	Internet address manipulation routines	inet(3N)
spline:	interpolate smooth curve	spline(1G)
siginterrupt: allow signals to	interrupt system calls	siginterrupt(3)
onintr: process intro:	interrupts in command scripts introduction to C library functions	csh(1) intro(3)
intro:	introduction to display graphics subroutines	intro(3G)
intro:	introduction to FORTRAN library functions	intro(3F)
math:	introduction to mathematical library functions	math(3M)
intro: intro:	introduction to special files and hardware support introduction to system calls and error numbers	intro(4) intro(2)
commands. intro:	introduction to system maintenance and operation	intro(8)
(temporary). infnan: signals	invalid floating-point operations on a VAX	infnan(3M)
asinh, acosh, atanh:	inverse hyperbolic functions	asinh(3M)
atan, atan2: trigonometric functions and their	inverses. sin, cos, tan, asin, acos,	sin(3M)
m_in, mout,/ madd, msub, mult, mdiv, pow, gcd, tread, twrite, trewin, tskipf, tstate: f77 tape	invert, rpow, msqrt, mcmp, move, min, omin, fmin,	mp(3X)
bus: control of access to the system	I/O topen, tclose,	topen(3F) bus(4)
ioinit: change f77	I/O initialization.	ioinit(3F)
popen, pclose: initiate	I/O to/from a process	popen(3)
nand Annud	ioinit: change f77 I/O initialization.	
rand, drand, random, drandm,	irand: return random values	rand(3F) random(3F)
isalpha, isupper, islower, isdigit, isxdigit,	isalnum, isspace, ispunct, isprint, isgraph,/	ctype(3)
isalnum, isspace, ispunct, isprint, isgraph,/	isalpha, isupper, islower, isdigit, isxdigit,	ctype(3)
/isspace, ispunct, isprint, isgraph, iscntrl,	isascii, toupper, tolower, toascii: character/	ctype(3)
ttynam,	isatty: find name of a terminal port.	ttynam(3F)
ttyname, /isalnum, isspace, ispunct, isprint, isgraph,	isatty, ttyslot: find name of a terminal iscntrl, isascii, toupper, tolower, toascii:/	ttyname(3) ctype(3)
isprint, isgraph,/ isalpha, isupper, islower,	isdigit, isadigit, isalnum, isspace, ispunct,	ctype(3)
/isxdigit, isalnum, isspace, ispunct, isprint,	isgraph, iscntrl, isascii, toupper, tolower,/	ctype(3)
ispunct, isprint, isgraph,/ isalpha, isupper,	islower, isdigit, isxdigit, isalnum, isspace,	ctype(3)
/drem, logb, scalb, rint, classdouble, classfloat,	isnan, unordered, finite, infinity, nextdouble,/	ieee(3)
/isdigit, isxdigit, isalnum, isspace, ispunct, /islower, isdigit, isxdigit, isalnum, isspace,	isprint, isgraph, isentrl, isascii, toupper,/ ispunct, isprint, isgraph, isentrl, isascii,/	ctype(3) ctype(3)
/isupper, islower, isdigit, isxdigit, isalnum,	isspace, ispunct, isprint, isgraph, iscntrl,/	ctype(3)
system:	issue a shell command	system(3)
isspace, ispunct, isprint, isgraph,/ isalpha,	tono or or toloring to distant and the toloring	ctype(3)
isgraph,/ isalpha, isupper, islower, isdigit, idate.	isupper, islower, isdigit, isxdigit, isalnum,	* * * * * * * * * * * * * * * * * * * *
IOAIP.	isxdigit, isalnum, isspace, ispunct, isprint,	ctype(3)
•	isxdigit, isalnum, isspace, ispunct, isprint, itime: return date or time in numerical form.	ctype(3) idate(3F)
omin, fmin, m_in, mout, omout, fmout, m_out, sdiv,	isxdigit, isalnum, isspace, ispunct, isprint, itime: return date or time in numerical form	ctype(3)

:0 :1	in and and some December Companies	:0/23/1
j0, j1,		j0(3M)
bg: place	• • • • • • • • • • • • • • • • • • • •	csh(1)
fg: bring	job list.	csh(1) csh(1)
stop: halt a kill: kill		csh(1) csh(1)
Kili. Kili	jobs: print current job list.	csh(1)
	kbdemul: default keyboard emulator.	kbdemul(4)
	kbdlock: lock the keyboard of the IBM RT PC	kbdlock(1)
bufemul:		bufemul(4)
cons:	keyboard and console display interface	cons(4)
xemul: X input emulator for queuing	keyboard and console display interface	xemul(4)
kbdemul: default	•	kbdemul(4)
mset: retrieve ASCII to IBM 3270	keyboard map.	mset(1)
xmodmap, xprkbd - X Window System	keyboard modifier utilities.	xmodmap(1)
kbdlock: lock the	keyboard of the IBM RT PC.	kbdlock(1)
pf: set		
keyboard codes:	keyboard scancode table.	keyboard codes(5)
kcyoodid_codes.	keyboard codes: keyboard scancode table	keyboard_codes(5)
base for mapping ASCII keystrokes into IBM 3270	keys. map3270: data	map3270(5)
pf: set keyboard program-function	keys	pf(1)
map3270: data base for mapping ASCII	keystrokes into IBM 3270 keys.	map3270(5)
kill:	kill jobs and processes.	csh(1)
Kill.	kill: kill jobs and processes.	csh(1)
	kill: send a signal to a process	kill(3F)
bessel functions: of two		
	kmem, kmem1, kmem2, kmem4, ros, afpamem: main	bessel(3F) mem(4)
memory. mem, mem. kmem.		, ,
	kmem1, kmem2, kmem4, ros, afpamem: main memory. kmem2, kmem4, ros, afpamem: main memory	mem(4)
mem, kmem, kmem1, mem, kmem, kmem1, kmem2,	kmem2, kmem4, ros, afpamem: main memory kmem4, ros, afpamem: main memory	mem(4)
	label, line, circle, arc, move, cont, point,	mem(4)
linemod, space, closepl:/ plot: openpl, erase,		plot(3X)
Adapter.	lan: IBM RT PC Token-Ring Adapter landump: dump IBM Token-Ring Personal Computer	lan(4) landump(8r)
Xlib: C	Language X Window System Interface Library	Xlib(3X)
Allo. C	ld: link editor	Id(1)
froun		
frexp,	Idexp, modf: split into mantissa and exponent learn: computer aided instruction about UNIX	frexp(3)
ta-	1	learn(1)
exit:		csh(1)
getusershell, setusershell, endusershell: get	legal user shells	getusershell(3)
index, rindex, Inblnk, truncate, ftruncate: truncate a file to a specified		index(3F)
	length	truncate(2) getopt(3)
getopt: get option rvdchlog: change logging	level of Remote Virtual Disk (RVD) server	rvdchlog(8)
rvdcniog: change logging	learner les comme function	
terminal.	lgamma: log gamma function lib2648: subroutines for the HP 2648 graphics	lgamma(3M) lib2648(3X)
et al.: f77 library interface to plot (3X)	libraries plot: openpl	plot(3F)
Xlib: C Language X Window System Interface	Library.	Xlib(3X)
intro: introduction to C	library functions.	intro(3)
intro: introduction to FORTRAN	library functions.	intro(3F)
math: introduction to mathematical	and the second s	math(3M)
	library interface to plot (3X) libraries.	plot(3F)
piot. openpi et al 177	limit: alter per-process resource limitations.	csh(1)
limit: alter per-process resource	limitations.	csh(1)
unlimit: remove resource	limitiations.	csh(1)
VI ALine, VI RLine: draw a	line.	line(3G)
getarg, iarge: return command	line arguments.	getarg(3F)
space, closepl:/ plot: openpl, erase, label,	line, circle, arc, move, cont, point, linemod,	plot(3X)
VI Dash: set	line dash pattern.	dash(3G)
ap: asynchronous data mode protocol	line discipline.	ap(4)
tb:	line discipline for digitizing devices.	tb(4)
lp:	line printer	lp(4)
VI_Width: set	line width.	width(3G)
/erase, label, line, circle, arc, move, cont, point,	linemod, space, closepl: graphics interface.	plot(3X)
ld:	link editor.	ld(1)
a.out: assembler and	link editor output.	a.out(5)
move apportion and	link: make a link to an existing file.	link(3F)
link: make a	link to an existing file.	link(3F)
glob: filename expand argument	list.	csh(1)
history: print history event	list.	csh(1)
jobs: print current job	list.	csh(1)
shift: manipulate argument	list.	csh(1)
initgroups: initialize group access	list.	initgroups(3)
nlist: get entries from name	list.	nlist(3)
varargs: variable argument	list	varargs(3)
vdstats:	list client Remote Virtual Disk (RVD) statistics	vdstats(8)
xlsfonts - X window system font	list displayer	xlsfonts(1)

foreach: loop over	list of names.	csh(1)
rvdhosts: print a	list of RVD servers.	
vgrind: grind nice	listings of programs for the IBM 3812 Pageprinter	vgrind(1)
index, rindex,	Inblnk, len: tell about character objects	
xload - X window system	load average display	xload(1)
Point Accelerator. afpacode:	loc: return the address of an object	afpacode(8r) loc(3F)
and time to ASCII. ctime,	localtime, gmtime, asctime, timezone: convert date	ctime(3)
end, etext, edata: last	locations in program	end(3)
kbdlock:	lock the keyboard of the IBM RT PC	kbdlock(1)
openlog, closelog, setlogmask: control system	log. syslog,	syslog(3)
aedjournal: display commands in a	log file	aedjournal(1) aedrunner(1)
aedrunner: execute graphics commands in a begin logging subroutine calls and close a	log file. VI Login, VI Logout:	log(3G)
VI Run: process a	log file.	run(3G)
lgamma:	log gamma function	lgamma(3M)
nfabort: dump core and	log it in a notesfile.	nfabort(3)
power. exp, expm1,	log, log10, log1p, pow: exponential, logarithm,	exp(3M)
rvdlog: cause Remote Virtual Disk (RVD) server to syslogd:	log statistics	rvdlog(8) syslogd(8)
exp, expm1, log,	log10, log1p, pow: exponential, logarithm, power.	exp(3M)
exp, expm1, log, log10,	log1p, pow: exponential, logarithm, power	exp(3M)
exp, expm1, log, log10, log1p, pow: exponential,	logarithm, power	exp(3M)
manipulations. copysign, drem, finite,	logb, scalb: copysign, remainder, exponent	ieee(3M)
unordered, finite, infinity,/ copysign, drem, rvdchlog: change	logb, scalb, rint, classdouble, classfloat, isnan, logging level of Remote Virtual Disk (RVD) server	ieee(3) rvdchlog(8)
VI Login, VI Logout: begin	logging subroutine calls and close a log file	log(3G)
flush: flush output to a	logical unit.	flush(3F)
fseek, ftell: reposition a file on a	logical unit.	fseek(3F)
getc, fgetc: get a character from a	logical unit.	getc(3F)
putc, fputc: write a character to a fortran	logical unit.	putc(3F)
getlog: get user's	login: login new user.	csh(1) getlog(3F)
getlogin: get	login name.	getlogin(3)
login:	login new user.	csh(1)
tn3270: full-screen remote	login to IBM VM/CMS	tn3270(1)
A 5	logout: end session.	csh(1)
setjmp, break: exit while/foreach	longjmp: non-local goto.	setjmp(3) csh(1)
continue: cycle in	loop	csh(1)
end: terminate	loop	csh(1)
foreach:	loop over list of names	csh(1)
11. 1	lp: line printer.	lp(4)
bit: and, or, xor, not, rshift, stat,	Ishift bitwise functions	bit(3F) stat(3F)
time, ctime,	ltime, gmtime: return system time.	time(3F)
getfloatstate: return	machine and process floating point state	getfloatstate(2)
alias: shell	macros	7 7
toupper, tolower, toascii: character classification	macros. /isprint, isgraph, iscntrl, isascii,	ctype(3)
msqrt, mcmp, move, min, omin, fmin, m_in, mout,/	madd, msub, mult, mdiv, pow, gcd, invert, rpow, magnetic tape manipulating program	mp(3X) mt(1)
mt: mtio: 4.3/RT	magtape interface.	
xmh: X window interface to the mh	Mail Handler.	xmh(1)
mem, kmem, kmem1, kmem2, kmem4, ros, afpamem:	main memory.	mem(4)
intro: introduction to system	maintenance and operation commands	intro(8)
fdisk: boot record partition table minidisk: minidisk	maintenance utility	fdisk(8) minidisk(8r)
link:	make a link to an existing file.	link(3F)
mktemp:	make a unique file name	mktemp(3)
makesym:	make debugger symbol table	makesym(8)
makedev:	make system special files.	makedev(8)
	makedev: make system special files	makedev(8) makesym(8)
	malloc, free, falloc: memory allocator	malloc(3F)
allocator.	malloc, free, realloc, calloc, alloca: memory	malloc(3)
vddb: Remote Virtual Disk (RVD) data base	manager	vddb(8)
wm: a simple real-estate-driven window	manager.	wm(1)
.PP uwm - Window	Manager Client Application of X .PP	uwm(1)
shift: VI_Font, VI_GetFont, VI_DropFont: select and	manipulate argument list	csh(1) font(3G)
mt: magnetic tape	manipulating program.	mt(1)
inet_Inaof, inet_netof: Internet address	manipulation routines. /inet_ntoa, inet_makeaddr,	inet(3N)
finite, logb, scalb: copysign, remainder, exponent	manipulations. copysign, drem,	ieee(3M)
frexp, ldexp, modf: split into	mantissa and exponent	frexp(3)
into IBM 3270 keys. map3270: data base for	map3270: data base for mapping ASCII keystrokes mapping ASCII keystrokes into IBM 3270 keys	map3270(5) map3270(5)
mapszio, data base for	mapping ASCII keystrokes into IBM 3270 keys	111ap3210(3)

vannalus aliannaa an diantau Gla annatian		
umask: change or display file creation functions.	mask	csh(1)
math: introduction to	mathematical library functions.	math(3M) math(3M)
vlimit: control	maximum system resource consumption	vlimit(3C)
/msub, mult, mdiv, pow, gcd, invert, rpow, msqrt,	mcmp, move, min, omin, fmin, m in, mout, omout,/	mp(3X)
min, omin, fmin, m in, mout,/ madd, msub, mult,	mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move,	mp(3X)
ibm5081, mpel – IBM 5081	Mega Pel Display interface.	ibm5081(4)
memory.	mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main	
mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main	memory.	mem(4)
malloc, free, realloc, calloc, alloca:	memory allocator.	malloc(3)
malloc, free, falloc:	memory allocator	malloc(3F)
valloc: aligned	memory allocator	valloc(3C)
dumpaed: dump aed display	memory as a binary file	dumpaed(1)
dumpapal6: dump apal6 display	memory as a binary file.	dumpapa16(1)
dumpapa8: dump apa8 display	memory as a binary file.	dumpapa8(1)
dumpapa8c: dump apa8c display	memory as a binary file	dumpapa8c(1)
abort: terminate abruptly with	memory image.	abort(3F)
core: format of	memory image file.	core(5)
vmstat: report virtual	memory statistics.	vmstat(1)
XMenu - X Deck of cards	Menu System.	XMenu(3X)
VI_Merge: set omerge:	merge mode	merge(3G) omerge(8)
rvdgetm: get operations	message from Remote Virtual Disk (RVD) server	rvdgetm(8)
rvdsetm: set operations	message on Remote Virtual Disk (RVD) server	rvdsetm(8)
error: analyze and disperse compiler error	messages	error(1)
syslogd: log systems	messages.	syslogd(8)
perror, sys errlist, sys nerr: system error	messages.	perror(3)
perror, gerror, ierrno: get system error	messages.	perror(3F)
psignal, sys siglist: system signal	messages	psignal(3)
xmh: X window interface to the	mh Mail Handler	xmh(1)
invert, rpow, msqrt, mcmp, move, min, omin, fmin,	m_in, mout, omout, fmout, m_out, sdiv, itom://gcd, .	mp(3X)
/mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move,	min, omin, fmin, m_in, mout, omout, fmout, m_out,/ .	mp(3X)
minidisk:	minidisk maintenance utility.	minidisk(8r)
	minidisk: minidisk maintenance utility	minidisk(8r)
	mktemp: make a unique file name	mktemp(3)
VI_Merge: set merge	mode.	merge(3G)
chmod: change	mode of a file.	chmod(3F)
chmod, fchmod: change	mode of file.	chmod(2)
ap: asynchronous data	mode protocol line discipline.	ap(4)
frexp, ldexp,	modf: split into mantissa and exponent	frexp(3)
xmodmap, xprkbd - X Window System keyboard	modifier utilities	xmodmap(1)
monitor, monstartup,	moncontrol: prepare execution profile	monitor(3) monitor(3)
profile. ibm5151,	monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface	ibm5151(4)
ibm5151, mono: IBM 5151	Monochrome Display interface	ibm5151(4)
ibm6153, apa8: IBM 6153 Advanced	Monochrome Graphics Display interface	ibm6153(4)
ibm6155, apa16: IBM 6155 Extended	Monochrome Graphics Display interface	ibm6155(4)
monitor.	monstartup, moncontrol: prepare execution profile	monitor(3)
curses: screen functions with "optimal" cursor	motion.	curses(3X)
xemul: X input emulator for queuing keyboard and	mouse events	xemul(4)
mouse:	mouse interface	mouse(4)
	mouse: mouse interface	mouse(4)
/rpow, msqrt, mcmp, move, min, omin, fmin, m_in,	mout, omout, fmout, m_out, sdiv, itom: multiple/	mp(3X)
/move, min, omin, fmin, m_in, mout, omout, fmout,	m_out, sdiv, itom: multiple precision integer/	mp(3X)
plot: openpl, erase, label, line, circle, arc,	move, cont, point, linemod, space, closepl:/	plot(3X)
/mult, mdiv, pow, gcd, invert, rpow, msqrt, mcmp,	move, min, omin, fmin, m_in, mout, omout, fmout,/	mp(3X)
VI_AMove, VI_RMove:	move the current point.	move(3G)
ibm5081,	mpel - IBM 5081 Mega Pel Display interface	ibm5081(4)
madd, msub, mult, mdiv, pow, gcd, invert, rpow,	mset: retrieve ASCII to IBM 3270 keyboard map msgrt, mcmp, move, min, omin, fmin, m in, mout./	mset(1)
mcmp, move, min, omin, fmin, m in, mout,/ madd,	msub, mult, mdiv, pow, gcd, invert, rpow, msqrt,	mp(3X) mp(3X)
monip, move, min, omin, min, m_in, mode, madd,	mt: magnetic tape manipulating program	mt(1)
	mtio: 4.3/RT magtape interface	mtio(4)
move, min, omin, fmin, m in, mout,/ madd, msub,	mult, mdiv, pow, gcd, invert, rpow, msqrt, mcmp,	mp(3X)
fmin, m in, mout, omout, fmout, m out, sdiv, itom:	multiple precision integer arithmetic. /min, omin,	mp(3X)
interface. asy:	multi-port asynchronous communications RS232C	asy(4)
switch:	multi-way command branch	csh(1)
getdiskbyname: get disk description by its	name	getdisk(3)
getenv: value for environment	name	getenv(3)
getlog: get user's login	name	getlog(3F)
getlogin: get login	name	getlogin(3)
mktemp: make a unique file	name.	mktemp(3)
getpw: get	name from uid.	getpw(3C)
nlist: get entries from ttyname, isatty, ttyslot: find	name list	nlist(3)
ttyname, isatty, ttysiot: find ttynam, isatty: find	name of a terminal port	ttyname(3)
ilynam, isaliy: nnd	name of a terminal port.	ttynam(3F)

hostnm: get	name of current host.	hostnm(3F)
foreach: loop over list of	names	
rvdexch: exchange restore.tape, restore.	net: install system from tape or over network	
restore.net: install system from tape or over	network. restore.tape,	
ntohl, ntohs: convert values between host and	network byte order. htonl, htons,	byteorder(3N)
getnetbyname, setnetent, endnetent: get	network entry. getnetent, getnetbyaddr,	getnetent(3N)
gethostent, sethostent, endhostent: get	network host entry. gethostbyname, gethostbyaddr, network interface parameters	gethostbyname(3N) ifconfig(8c)
ifconfig: configure newfs: construct a	new file system.	
newvd: create a	new filesystem on a Remote Virtual Disk (RVD)	
login: login	new user.	csh(1)
	newfs: construct a new file system	
Disk (RVD).	newvd: create a new filesystem on a Remote Virtual	
/classfloat, isnan, unordered, finite, infinity, /isnan, unordered, finite, infinity, nextdouble,	nextdouble, nextfloat, fptestround, fpsetround,/ nextfloat, fptestround, fpsetround, swapround,/	
dbminit, fetch, store, delete, firstkey,	nextkey: data base subroutines	
, , , , , , , , , , , , , , , , , , , ,	nfabort: dump core and log it in a notesfile	
system.	nscomment: a user interface to the notesfile	
Pageprinter. vgrind: grind	nice listings of programs for the IBM 3812	U ()
	nice: run low priority process	csh(1) nice(3C)
	nlist: get entries from name list.	
	nohup: run command immune to hangups	
setjmp, longjmp:	non-local goto.	setjmp(3)
bit: and, or, xor,	not, rshift, Ishift bitwise functions	
nfabort: dump core and log it in a nfcomment: a user interface to the	notesfile	
notify: request immediate	notification.	· · · · · · · · · · · · · · · · · · ·
noinj, roquest miniosiate	notify: request immediate notification.	
Graphics Printer. proff:	nroff for the IBM 4201 Proprinter and IBM 5152	proff(1)
prfl: IBM 4201 Proprinter/IBM 5152 Graphics Printer	nroff post-processing filter	
routines. ns addr,	ns_addr, ns_ntoa: Xerox NS(tm) address conversion ns_ntoa: Xerox NS(tm) address conversion routines	
ns_addr, ns_ntoa: Xerox	NS(tm) address conversion routines	
network byte order. htonl, htons,	ntohl, ntohs: convert values between host and	
order. htonl, htons, ntohl,	ntohs: convert values between host and network byte .	byteorder(3N)
rand, srand: random	number generator	
random, drandm, irandm: better random	number generator.	random(3F)
random, srandom, initstate, setstate: better random intro: introduction to system calls and error	number generator; routines for changing generators. numbers	
atof, atoi, atol: convert ASCII to	numbers.	atof(3)
idate, itime: return date or time in	numerical form.	
loc: return the address of an	object.	loc(3F)
long, short: integer	object conversion.	
omerge: merge index, rindex, lnblnk, len: tell about character	object files	
midex, imidex, montk, tent ten about character	omerge: merge object files.	
/pow, gcd, invert, rpow, msqrt, mcmp, move, min,	omin, fmin, m_in, mout, omout, fmout, m_out, sdiv,/	mp(3X)
/msqrt, mcmp, move, min, omin, fmin, m_in, mout,	omout, fmout, mout, sdiv, itom: multiple precision/	
rvdcopy: copy contents of	one RVD disk pack to another	rvdcopy(8)
afpacode: load, test, and bring	onintr: process interrupts in command scripts online the Advanced Floating Point Accelerator	
fopen, freopen, fdopen:	open a stream.	
closedir: directory operations.	opendir, readdir, telldir, seekdir, rewinddir,	
syslog,	openlog, closelog, setlogmask: control system log	syslog(3)
cont, point, linemod, space, closepl:/ plot:	openpl, erase, label, line, circle, arc, move, openpl et al.: f77 library interface to plot	plot(3X)
(3X) libraries plot: intro: introduction to system maintenance and	openpl et al.: f77 library interface to plot	plot(3F) intro(8)
tgetstr, tgoto, tputs: terminal independent	operation routines. tgetent, tgetnum, tgetflag,	termcap(3X)
bcopy, bcmp, bzero, ffs: bit and byte string	operations.	bstring(3)
telldir, seekdir, rewinddir, closedir: directory	operations. opendir, readdir,	
strcpy, strncpy, strlen, index, rindex: string	operations. strcat, strcat, strcmp, strncmp, operations message from Remote Virtual Disk (RVD)	string(3) rvdgetm(8)
server. rvdgetm: get server. rvdsetm: set	operations message from Remote Virtual Disk (RVD) operations message on Remote Virtual Disk (RVD) .	rvdsetm(8)
infnan: signals invalid floating-point	operations on a VAX (temporary)	infnan(3M)
curses: screen functions with	"optimal" cursor motion	, ,
getopt: get	option letter from argv	U 1 1 7
ntohs: convert values between host and network byte bessel functions: of two kinds for integer	order. htonl, htons, ntohl,	byteorder(3N) bessel(3F)
VI Force: force output of graphics	orders.	• •
a.out: assembler and link editor	output.	a.out(5)
terminate a process after flushing any pending	output. exit:	exit(3)
ecvt, fcvt, gcvt:	output conversion	ecvt(3)
printf, fprintf, sprintf: formatted	output conversion	printf(3S) stdemul(4)

5152 Graphics Printer. ibmbit, ibmgra, ibmpro:	output filters for the IBM 4201 Proprinter and IBM .	lpfilter(8r)
VI_Force: force	output of graphics orders.	force(3G)
flush: flush Xtext: routines to provide simple text	output to a logical unit	flush(3F) xtext(3X)
foreach: loop	over list of names.	csh(1)
restore.net: install system from tape or	over network. restore.tape,	restore.tape(8)
trapov: trap and repair floating point	overflow	trapov(3F)
exec: chown, fchown: change	overlay shell with specified command	csh(1) chown(2)
force spindown of a Remote Virtual Disk (RVD)	pack. rvddown:	rvddown(8)
spindown: spin up/down Remote Virtual Disk (RVD)	pack. spinup,	spinup(8)
rvdcopy: copy contents of one RVD disk	pack to another.	rvdcopy(8)
format: how to format disk	packs	format(8V)
exchange names of two Remote Virtual Disk (RVD) spindown client's Remote Virtual Disk (RVD)	packs. rvdflush:	rvdexch(8) rvdflush(8)
back up and restore Remote Virtual Disk (RVD)	packs to and from tape. savervd, zaprvd, savephys:	savervd(8)
3820 and IBM 3800 fonts for use with the IBM 3812	Pageprinter. /cvt20to12, cvt00to12: convert IBM	cvt3812(8)
pprint: print text files on IBM 3812	Pageprinter	pprint(1)
ppt: spooling system filter for the IBM 3812 ptroff: print troff files on IBM 3812	Pageprinter	ppt(8) ptroff(1)
grind nice listings of programs for the IBM 3812	Pageprinter. vgrind:	vgrind(1)
width3812: build width tables for IBM 3812	Pageprinter fonts	width3812(8)
ibm3812pp: IBM 3812	Pageprinter server	ibm3812pp(8)
printer3812: IBM 3812 xsetroot: X window system root window	Pageprinter status information	printer3812(5) xsetroot(1)
ifconfig: configure network interface	parameters	ifconfig(8c)
VI_QMerge, VI_QPoint, VI_QWidth: query graphics	parameters. /VI_QColor, VI_QDash, VI_QFont,	query(3G)
diskpart: calculate default disk	partition sizes	diskpart(8)
fdisk: boot record	partition table maintenance utility	fdisk(8)
pp: Professional getpass: read a	Pascal compiler	pp(1) getpass(3)
getpwnam, setpwent, endpwent, setpwfile: get	password file entry. getpwent, getpwuid,	getpuss(3)
getwd: get current working directory	pathname	getwd(3)
getcwd: get	pathname of current working directory	getcwd(3F)
VI_Dash: set line dash	pattern	dash(3G)
debug: debugger for the IBM RT	PC	pause(3C) debug(8)
kbdlock: lock the keyboard of the IBM RT	PC	kbdlock(1)
un: IBM RT	PC Baseband Adapter for use with Ethernet	un(4)
lan: IBM RT	PC Token-Ring Adapter	lan(4)
pcc:	pcc-based C compiler	pcc(1) pcc(1)
dosread: read, write, dir, delete on	PC-DOS diskette	dosread(1)
popen,	pclose: initiate I/O to/from a process	popen(3)
ibm5081, mpel – IBM 5081 Mega	Pel Display interface	ibm5081(4)
exit: terminate a process after flushing any limit: alter	per-process resource limitations.	exit(3) csh(1)
	perror, gerror, ierrno: get system error messages	perror(3F)
messages.	perror, sys_errlist, sys_nerr: system error	
landump: dump IBM Token-Ring	Personal Computer Adapter	landump(8r)
pictures.	pf: set keyboard program-function keys	pf(1) pic(1)
pic: troff preprocessor for drawing simple	pictures.	pic(1)
bg:	place job in background.	csh(1)
psp:	planar serial port RS232C interface	psp(4) plot(1G)
move, cont, point, linemod, space, closepl:/	plot: openpl, erase, label, line, circle, arc,	plot(3X)
plot (3X) libraries	plot: openpl et al.: f77 library interface to	plot(3F)
VI_AMove, VI_RMove: move the current	point.	move(3G)
load, test, and bring online the Advanced Floating fpa: direct interface to floating	Point Accelerator. afpacode:	afpacode(8r) fpa(3X)
trpfpe, fpecnt: trap and repair floating	point faults.	trpfpe(3F)
/erase, label, line, circle, arc, move, cont,	point, linemod, space, closepl: graphics interface	plot(3X)
trapov: trap and repair floating	point overflow.	trapov(3F)
getfloatstate: return machine and process floating	point state	getfloatstate(2)
popd:	pop shell directory stack	csh(1) csh(1)
	popen, pclose: initiate I/O to/from a process.	popen(3)
ttynam, isatty: find name of a terminal	port	ttynam(3F)
psp: planar serial	port RS232C interface	psp(4)
IBM 4201 Proprinter/IBM 5152 Graphics Printer nroff exp, expm1, log, log10, log1p,	post-processing filter. prfl:	prfl(1) exp(3M)
omin, fmin, m in, mout,/ madd, msub, mult, mdiv,	pow, gcd, invert, rpow, msqrt, mcmp, move, min,	mp(3X)
log, log10, log1p, pow: exponential, logarithm,	power. exp, expm1,	exp(3M)
.PP uwm - Window Manager Client Application of X.	PP.	uwm(1)
	pp: Professional Pascal compiler	pp(1)

.PP	PP uwm - Window Manager Client Application of X .	uwm(1)
.rr	pprint: print text files on IBM 3812 Pageprinter	pprint(1)
Pageprinter.	ppt: spooling system filter for the IBM 3812	ppt(8)
mout, omout, fmout, m_out, sdiv, itom: multiple	precision integer arithmetic. /omin, fmin, m in,	mp(3X)
monitor, monstartup, moncontrol:	prepare execution profile	monitor(3)
pic: troff	preprocessor for drawing simple pictures	pic(1)
nroff post-processing filter.	prfl: IBM 4201 Proprinter/IBM 5152 Graphics Printer .	prfl(1)
rvdhosts:	print a list of RVD servers	rvdhosts(8)
date:	print and set the date	date(1)
hashstat:	print command hashing statistics	csh(1)
jobs:	print current job list	csh(1)
history:	print history event list	csh(1)
bitprt: capture the image on a bitmap display and	print it on an IBM printer	bitprt(1)
pstat:	print system facts	pstat(8)
pprint:	print text files on IBM 3812 Pageprinter	pprint(1)
ptroff:	print troff files on IBM 3812 Pageprinter	ptroff(1)
xpr:	print X window dump	xpr(1)
image on a bitmap display and print it on an IBM	printer. bitprt: capture the	bitprt(1)
lp: line	printer	lp(4)
for the IBM 4201 Proprinter and IBM 5152 Graphics	Printer. ibmbit, ibmgra, ibmpro: output filters	lpfilter(8r)
for the IBM 4201 Proprinter and IBM 5152 Graphics	Printer. proff: nroff	proff(1)
prfl: IBM 4201 Proprinter/IBM 5152 Graphics	Printer nroff post-processing filter	prfl(1)
information.	printer3812: IBM 3812 Pageprinter status	printer3812(5)
conversion.	printf, fprintf, sprintf: formatted output	printf(3S)
nice: set program	priority.	nice(3C)
nice: run low	priority process	csh(1)
reboot: bootstrapping	procedures	reboot(8)
nice: run low priority	process	csh(1)
stop: halt a job or	process.	csh(1)
fork: create a copy of this	process.	fork(3F)
kill: send a signal to a	process	kill(3F)
popen, pclose: initiate I/O to/from a	process.	popen(3)
VI_Run:	process a log file.	run(3G)
exit: terminate a	process after flushing any pending output.	exit(3)
init:	process control initialization.	init(8)
getfloatstate: return machine and	process floating point state	getfloatstate(2)
getpid: get	process id	getpid(3F)
onintr:	process interrupts in command scripts	csh(1)
times: get	process times.	times(3C)
wait: wait for a	process to terminate	wait(3F)
ptrace:	process trace.	ptrace(2)
exit: terminate	process with status	exit(3F)
kill: kill jobs and	processes	csh(1)
wait: wait for background	processes to complete	csh(1)
halt: stop the	processor.	halt(8)
pp:	Professional Pascal compiler.	pp(1)
5152 Graphics Printer.	proff: nroff for the IBM 4201 Proprinter and IBM	proff(1)
monitor, monstartup, moncontrol: prepare execution	profile	monitor(3)
mt: magnetic tape manipulating	program	mt(1)
syscall: system call interface	program.	syscall(8)
end, etext, edata: last locations in		end(3)
xhost - X window system access control xset - X window system user setup	program.	xhost(1)
nice: set	program priority.	xset(1) nice(3C)
ptfinstall: install a	Program Temporary Fix (PTF)	
pumstan assert:	program verification	assert(3)
pf: set keyboard	program-function keys	pf(1)
vgrind: grind nice listings of	programs for the IBM 3812 Pageprinter	vgrind(1)
xprop - X Window System	property displayer	xprop(1)
colpro: column filter for IBM 4201	Proprinter	colpro(1)
ibmgra, ibmpro: output filters for the IBM 4201	Proprinter and IBM 5152 Graphics Printer. ibmbit,	lpfilter(8r)
proff: nroff for the IBM 4201	Proprinter and IBM 5152 Graphics Printer	proff(1)
post-processing filter. prfl: IBM 4201	Proprinter/IBM 5152 Graphics Printer nroff	prfl(1)
rvd: Remote Virtual Disk	protocol.	rvd(4p)
getprotobyname, setprotoent, endprotoent: get	protocol entry. getprotoent, getprotobynumber,	getprotoent(3N)
ap: asynchronous data mode	protocol line discipline.	ap(4)
Xtext: routines to	provide simple text output windows	xtext(3X)
Xtty: routines to	provide terminal emulator windows	xtty(3X)
	psignal, sys siglist: system signal messages	psignal(3)
	psp: planar serial port RS232C interface	psp(4)
	pstat: print system facts	pstat(8)
ptfinstall: install a Program Temporary Fix	(PTF)	ptfinstall(8)
	ptfinstall: install a Program Temporary Fix (PTF)	ptfinstall(8)
	ptrace: process trace	ptrace(2)
	ptroff: print troff files on IBM 3812 Pageprinter	ptroff(1)
ungetc:	ptroff: print troff files on IBM 3812 Pageprinter push character back into input stream	ptroff(1) ungetc(3S)

pushd:	push shell directory stack	csh(i)
*	pushd: push shell directory stack	csh(1)
puts, fputs:	put a string on a stream.	
putc, putchar, fputc, putw: unit.	put character or word on a stream.	· \
on a stream.	putc, fputc: write a character to a fortran logical putc, putchar, fputc, putw: put character or word	putc(3F) putc(3S)
stream. putc,	putchar, fputc, putw. put character or word on a	putc(3S)
	puts, fputs: put a string on a stream	
putc, putchar, fputc,	putw: put character or word on a stream	putc(3S)
	qsort: quick sort.	• :
WI OF THE ME ON THE ORIGINAL AND ONE RELEASE	qsort: quicker sort.	• • • • • • • • • • • • • • • • • • • •
VI_QFont, VI_QMerge, VI_QPoint, VI_QWidth: insque, remque: insert/remove element from a	query graphics parameters. /VI_QColor, VI_QDash, queue	query(3G)
xemul: X input emulator for	queuing keyboard and mouse events	insque(3) xemul(4)
qsort:	quick sort	qsort(3F)
qsort:	quicker sort.	
•	rand, drand, irand: return random values	rand(3É)
	rand, srand: random number generator	rand(3C)
generator.	random, drandm, irandm: better random number	random(3F)
rand, srand:	random number generator	rand(3C)
random, drandm, irandm: better random, srandom, initstate, setstate: better	random number generator	random(3F) random(3)
number generator; routines for changing/	random, srandom, initstate, setstate: better random	random(3)
rand, drand, irand: return	random values.	rand(3F)
,,	rc.config: configuration file for startup scripts	rc.config(5)
stream to a remote command.	rcmd, rresvport, ruserok: routines for returning a	rcmd(3)
tbuffer: streaming tape buffered	read	tbuffer(8)
getpass:	read a password.	getpass(3)
source:	read commands from file	csh(1)
VI_MRead, VI_FRead: dosread:	read display data	read(3G)
directory operations. opendir,	readdir, telldir, seekdir, rewinddir, closedir:	dosread(1) directory(3)
wm: a simple	real-estate-driven window manager	wm(1)
malloc, free,	realloc, calloc, alloca: memory allocator	malloc(3)
	reboot: bootstrapping procedures	reboot(8)
	re_comp, re_exec: regular expression handler	regex(3)
rehash:	recompute command hash table	csh(1)
fdisk: boot	record partition table maintenance utility	fdisk(8)
VI_Tile: tile a eval:	rectangle	tile(3G) csh(1)
re comp,	re exec: regular expression handler.	regex(3)
xrefresh -	refresh all windows on the screen	xrefresh(1)
re_comp, re_exec:	regular expression handler	regex(3)
	rehash: recompute command hash table	csh(1)
copysign, drem, finite, logb, scalb: copysign,	remainder, exponent manipulations	ieee(3M)
ruserok: routines for returning a stream to a rexec: return stream to a	remote command. rcmd, rresvport,	rcmd(3) rexec(3)
tn3270: full-screen	remote login to IBM VM/CMS	tn3270(1)
rvd:	Remote Virtual Disk protocol.	rvd(4p)
newvd: create a new filesystem on a		newvd(8)
vdspin, vdspind: spin up or spin down a	Remote Virtual Disk (RVD)	vdspin(2)
vddb:	Remote Virtual Disk (RVD) data base manager	vddb(8)
rvddown: force spindown of a	Remote Virtual Disk (RVD) pack	rvddown(8)
spinup, spindown: spin up/down rvdexch: exchange names of two	Remote Virtual Disk (RVD) pack	spinup(8) rvdexch(8)
rvdexch; exchange names of two rvdflush; spindown client's	Remote Virtual Disk (RVD) packs	rvdexch(8)
savervd, zaprvd, savephys: back up and restore	Remote Virtual Disk (RVD) packs to and from tape	savervd(8)
rvdchlog: change logging level of	Remote Virtual Disk (RVD) server	rvdchlog(8)
rvdgetm: get operations message from	Remote Virtual Disk (RVD) server	rvdgetm(8)
rvdsend - send control stream to	Remote Virtual Disk (RVD) server	rvdsend(8)
rvdsetm: set operations message on	Remote Virtual Disk (RVD) server	rvdsetm(8)
rydshow: show connections to	Remote Virtual Disk (RVD) server	rvdshow(8)
rvdshut: force shutdown of table. rvddb:	Remote Virtual Disk (RVD) server	rvdshut(8) rvddb(5)
rvdsrv:	Remote Virtual Disk (RVD) server daemon	rvdsrv(8)
rvdlog: cause	Remote Virtual Disk (RVD) server to log statistics	rvdiog(8)
vdstats: acquire client	Remote Virtual Disk (RVD) statistics	vdstats(2)
vdstats: list client	Remote Virtual Disk (RVD) statistics	vdstats(8)
up, down: client	Remote Virtual Disk (RVD) utilities	up(1)
/etc/rvd/rvdtab: information about client	Remote Virtual Disks (RVDs)	rvdtab(5)
unlink: unalias:	remove a directory entry	unlink(3F)
unsetenv:	remove environment variables.	csh(1) csh(1)
unscent. unlimit:	remove resource limitiations.	csh(1)
insque,	remque: insert/remove element from a queue	insque(3)
rename:	rename a file.	rename(3F)

	rename: rename a file.	rename(3F)
trpfpe, fpecnt: trap and	repair floating point faults.	trpfpe(3F)
trapov: trap and	repair floating point overflow	trapov(3F)
while:	repeat commands conditionally	csh(1)
	repeat: execute command repeatedly	
repeat: execute command	repeatedly	csh(1)
•	report virtual memory statistics.	
vmstat:		
fseek, ftell:	reposition a file on a logical unit.	fseek(3F)
fseek, ftell, rewind:	reposition a stream	fseek(3S)
notify:	request immediate notification.	csh(1)
disk: format of	reserved areas of the hard disk	
res mkquery, res send,	res_init, dn_comp, dn_expand: resolver routines	
scale:	resize a bitmap image.	
dn_expand: resolver routines.	res_mkquery, res_send, res_init, dn_comp,	
res_send, res_init, dn_comp, dn_expand:	resolver routines. res_mkquery,	resolver(3)
vlimit: control maximum system	resource consumption	vlimit(3C)
xrdb - Server	Resource Database Utility	xrdb(i)
limit: alter per-process	resource limitations.	csh(1)
unlimit: remove	resource limitiations	csh(1)
vtimes: get information about	resource utilization.	
routines. res_mkquery,	res_send, res_init, dn_comp, dn_expand: resolver	resolver(3)
restore: incremental file system	restore	restore(8)
· ·	restore: incremental file system restore	restore(8)
tape. savervd, zaprvd, savephys: back up and	restore Remote Virtual Disk (RVD) packs to and from	savervd(8)
	, , ,	· · ·
network. restore.tape,	restore.net: install system from tape or over	
or over network.	restore.tape, restore.net: install system from tape	restore.tape(8)
suspend: suspend a shell,	resuming its superior	csh(1)
mset:	retrieve ASCII to IBM 3270 keyboard map	mset(1)
getfpemulator:	return address of the floating-point emulator	getfpemulator(2)
getarg, iarge:	return command line arguments	
0 0.	return date and time in an ASCII string	fdate(3F)
fdate:	· · · · · · · · · · · · · · · · · · ·	
idate, itime:	return date or time in numerical form	idate(3F)
etime, dtime:	return elapsed execution time	etime(3F)
flmin, flmax, ffrac, dflmin, dflmax, dffrac, inmax:	return extreme values	flmin(3F)
getfloatstate:	return machine and process floating point state	getfloatstate(2)
rand, drand, irand:	return random values	rand(3F)
	return stream to a remote command	* *
rexec:		rexec(3)
time, ctime, ltime, gmtime:	return system time.	time(3F)
loc:	return the address of an object	loc(3F)
rcmd, rresvport, ruserok: routines for	returning a stream to a remote command	rcmd(3)
fseek, ftell,	rewind: reposition a stream	fseek(3S)
opendir, readdir, telldir, seekdir,	rewinddir, closedir: directory operations	
openan, readan, tenen, beatan,	rexec: return stream to a remote command	
index,	rindex, Inblnk, len: tell about character objects	1.5
strcmp, strncmp, strcpy, strncpy, strlen, index,	rindex: string operations. strcat, strncat,	
finite, infinity,/ copysign, drem, logb, scalb,	rint, classdouble, classfloat, isnan, unordered,	ieee(3)
cbrt, sqrt: cube root, square	root	sqrt(3M)
cbrt, sqrt: cube	root, square root.	sqrt(3M)
xsetroot: X window system	root window parameter setting utility	
mem, kmem, kmem1, kmem2, kmem4,	ros, afpamem: main memory.	
		incili(4)
inet netor: Internet address manipulation	routines. /inet_ntoa, inet_makeaddr, inet_lnaof,	inet(3N)
ns_addr, ns_ntoa: Xerox NS(tm) address conversion		ns(3N)
res_send, res_init, dn_comp, dn_expand: resolver	routines. res_mkquery,	
tgoto, tputs: terminal independent operation	routines. tgetent, tgetnum, tgetflag, tgetstr,	termcap(3X)
setstate: better random number generator;	routines for changing generators. /initstate,	random(3)
command. rcmd, rresvport, ruserok:	routines for returning a stream to a remote	rcmd(3)
Xtext:	routines to provide simple text output windows	xtext(3X)
Xtty:	routines to provide simple text output windows	xtty(3X)
		7.1
mout,/ madd, msub, mult, mdiv, pow, gcd, invert,	rpow, msqrt, mcmp, move, min, omin, fmin, m_in,	mp(3X)
to a remote command. rcmd,	rresvport, ruserok: routines for returning a stream	rcmd(3)
asy: multi-port asynchronous communications	RS232C interface	asy(4)
psp: planar serial port	RS232C interface	psp(4)
bit: and, or, xor, not,	rshift, Ishift bitwise functions	bit(3F)
debug: debugger for the IBM	RT PC.	debug(8)
kbdlock: lock the keyboard of the IBM	RT PC.	kbdlock(1)
· · · · · · · · · · · · · · · · · · ·		
un: IBM	RT PC Baseband Adapter for use with Ethernet	un(4)
lan: IBM	RT PC Token-Ring Adapter	lan(4)
nohup:	run command immune to hangups	csh(1)
nice:	run low priority process	csh(1)
remote command. rcmd, rresvport,	ruserok: routines for returning a stream to a	rcmd(3)
create a new filesystem on a Remote Virtual Disk	(RVD). newvd:	newvd(8)
	•	
vdspind: spin up or spin down a Remote Virtual Disk	(RVD). vdspin,	vdspin(2)
vddb: Remote Virtual Disk	(RVD) data base manager	vddb(8)
rvdcopy: copy contents of one	RVD disk pack to another	rvdcopy(8)
rvddown: force spindown of a Remote Virtual Disk	(RVD) pack	rvddown(8)
spinup, spindown: spin up/down Remote Virtual Disk	(RVD) pack	spinup(8)

	(BVD) mades	and analy (0)
rvdexch: exchange names of two Remote Virtual Disk		
rvdflush: spindown client's Remote Virtual Disk	(RVD) packs	rvdflush(8)
savephys: back up and restore Remote Virtual Disk		
shows Issuing Issuing of Bourse Vietnal Dist.	rvd: Remote Virtual Disk protocol.	
	(RVD) server. rvdchlog:	
get operations message from Remote Virtual Disk	(RVD) server. rvdgetm:	
- send control stream to Remote Virtual Disk	(RVD) server. rvdsend	
set operations message on Remote Virtual Disk	(RVD) server. rvdsetm:	
rvdshow: show connections to Remote Virtual Disk	(RVD) server	rvdshow(8)
rvdshut: force shutdown of Remote Virtual Disk	(RVD) server	
rvddb: Remote Virtual Disk	(RVD) server configuration table	
rvdsrv: Remote Virtual Disk	(RVD) server daemon	
rvdlog: cause Remote Virtual Disk	(RVD) server to log statistics	
rvdhosts: print a list of	RVD servers	
vdstats: acquire client Remote Virtual Disk	(RVD) statistics	vdstats(2)
vdstats: list client Remote Virtual Disk	(RVD) statistics	vdstats(8)
up, down: client Remote Virtual Disk	(RVD) utilities	
Disk (RVD) server.	rvdchlog: change logging level of Remote Virtual	rvdchlog(8)
another.	rvdcopy: copy contents of one RVD disk pack to	rvdcopy(8)
configuration table.	rvddb: Remote Virtual Disk (RVD) server	
(RVD) pack.	rvddown: force spindown of a Remote Virtual Disk	
(RVD) packs.	rvdexch: exchange names of two Remote Virtual Disk .	
(RVD) packs.	rvdflush: spindown client's Remote Virtual Disk	
Disk (RVD) server.	rvdgetm: get operations message from Remote Virtual .	
Disk (RVD) sciver.	rvdhosts: print a list of RVD servers	
lan etation		
log statistics.	rvdlog: cause Remote Virtual Disk (RVD) server to	
information about client Remote Virtual Disks	(RVDs). /etc/rvd/rvdtab:	
Disk (RVD) server.	rvdsend - send control stream to Remote Virtual	
Disk (RVD) server.	rvdsetm: set operations message on Remote Virtual	
(RVD) server.	rvdshow: show connections to Remote Virtual Disk	
(RVD) server.	rvdshut: force shutdown of Remote Virtual Disk	
	rvdsrv: Remote Virtual Disk (RVD) server daemon	
	rxformat: format floppy disks	
•	sautil: standalone utility package.	
(RVD) packs to and from tape. savervd, zaprvd,	savephys: back up and restore Remote Virtual Disk .	savervd(8)
Remote Virtual Disk (RVD) packs to and from tape.	savervd, zaprvd, savephys: back up and restore	savervd(8)
System Interface (SCSI) Adapter.	sc: IBM 9332 disks using the IBM Small Computer	sc(4)
copysign, drem, finite, logb,	scalb: copysign, remainder, exponent manipulations	ieee(3M)
unordered, finite, infinity,/ copysign, drem, logb,	scalb, rint, classdouble, classfloat, isnan,	
	and a material transmitters a	enele(1)
	scale: resize a bitmap image	scale(1)
scandir, alphasort:	scan a directory.	
scandir, alphasort: keyboard codes: keyboard		scandir(3)
	scan a directory.	scandir(3) keyboard_codes(5)
	scan a directory	scandir(3) keyboard_codes(5) scandir(3) scanf(3S)
	scan a directory	scandir(3) keyboard_codes(5) scandir(3) scanf(3S)
keyboard_codes: keyboard	scan a directory	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C)
keyboard_codes: keyboard alarm:	scan a directory	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen screen access. screen color. screen functions with "optimal" cursor motion. screens.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command rc.config: configuration file for startup	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. scripts. (SCSI) Adapter. sc: IBM 9332	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command rc.config: configuration file for startup disks using the IBM Small Computer System Interface	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command rc.config: configuration file for startup disks using the IBM Small Computer System Interface /min, omin, fmin, m_in, mout, omout, fmout, m_out,	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X)
keyboard_codes: keyboard alarm:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8)
keyboard_codes: keyboard alarm:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3)
keyboard_codes: keyboard alarm:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations. select and manipulate fonts.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command rc.config: configuration file for startup disks using the IBM Small Computer System Interface /min, omin, fmin, m_in, mout, omout, fmout, m_out, badsect: create files to contain bad opendir, readdir, telldir, VI_Font, VI_GetFont, VI_DropFont: case:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. scekdir, rewinddir, closedir: directory operations. select and manipulate fonts. selector in switch.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1)
keyboard_codes: keyboard alarm:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command rc.config: configuration file for startup disks using the IBM Small Computer System Interface /min, omin, fmin, m_in, mout, omout, fmout, m_out, badsect: create files to contain bad opendir, readdir, telldir, VI_Font, VI_GetFont, VI_DropFont: case: Academic Information Systems experimental display kill:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM send a signal to a process.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F)
keyboard_codes: keyboard alarm:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM send a signal to a process. send APAR.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command rc.config: configuration file for startup disks using the IBM Small Computer System Interface /min, omin, fmin, m_in, mout, omout, fmout, m_out, badsect: create files to contain bad opendir, readdir, telldir, VI_Font, VI_GetFont, VI_DropFont: case: Academic Information Systems experimental display kill:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM send a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD)	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8)
keyboard_codes: keyboard alarm:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM send a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD) sendapar: send APAR.	scandir(3) keyboard_codes(5) scandir(3) scand(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8) sendapar(8)
keyboard_codes: keyboard alarm:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations. select and manipulate fonts. select and a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD) sendapar: send APAR. serial port RS232C interface.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8) sendapar(8) psp(4)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command rc.config: configuration file for startup disks using the IBM Small Computer System Interface /min, omin, fmin, m_in, mout, omout, fmout, m_out, badsect: create files to contain bad opendir, readdir, telldir, VI_Font, VI_GetFont, VI_DropFont: case: Academic Information Systems experimental display kill: sendapar: server. rvdsend - psp: planar ibm3812pp: IBM 3812 Pageprinter	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. scetors. sceekdir, rewinddir, closedir: directory operations. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM send a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD) sendapar: send APAR. serial port RS232C interface.	scandir(3) keyboard_codes(5) scandir(3) scand(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8) sendapar(8)
keyboard_codes: keyboard alarm:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. sceekdir, rewinddir, closedir: directory operations. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM send a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD) sendapar: send APAR. serial port RS232C interface. server. server. rvdchlog:	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8) sendapar(8) psp(4)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command rc.config: configuration file for startup disks using the IBM Small Computer System Interface /min, omin, fmin, m_in, mout, omout, fmout, m_out, badsect: create files to contain bad opendir, readdir, telldir, VI_Font, VI_GetFont, VI_DropFont: case: Academic Information Systems experimental display kill: sendapar: server. rvdsend - psp: planar ibm3812pp: IBM 3812 Pageprinter	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. scetors. sceekdir, rewinddir, closedir: directory operations. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM send a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD) sendapar: send APAR. serial port RS232C interface.	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8) sendapar(8) psp(4) ibm3812pp(8)
keyboard_codes: keyboard alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command rc.config: configuration file for startup disks using the IBM Small Computer System Interface /min, omin, fmin, m_in, mout, omout, fmout, m_out, badsect: create files to contain bad opendir, readdir, telldir, VI_Font, VI_GetFont, VI_DropFont: case: Academic Information Systems experimental display kill: sendapar: server. rvdsend - psp: planar ibm3812pp: IBM 3812 Pageprinter change logging level of Remote Virtual Disk (RVD)	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. sceekdir, rewinddir, closedir: directory operations. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM send a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD) sendapar: send APAR. serial port RS232C interface. server. server. rvdchlog:	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8) sendapar(8) psp(4) ibm3812pp(8) rvdchlog(8)
keyboard_codes: keyboard alarm:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations. select and manipulate fonts. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM send a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD) sendapar: send APAR. serial port RS232C interface. server. server. rvdchlog: server. rvdgetm: get	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8) sendapar(8) rvdsend(8) sendapar(8) rvdchlog(8) rvdgetm(8)
keyboard_codes: keyboard alarm:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations. select and manipulate fonts. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM send a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD) sendapar: send APAR. serial port RS232C interface. server. server. rvdchlog: server. rvdchlog: server. rvdsend	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8) sendapar(8) rvdsend(8) rvdgetm(8) rvdgetm(8) rvdsend(8)
keyboard_codes: keyboard alarm:	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations. select and manipulate fonts. select or in switch. self-tests. aedtest: IBM send a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD) sendapar: send APAR. serial port RS232C interface. server. server. rvdchlog: server. rvdsed server. rvdsed server. rvdsetm: server. rvdsetm: server. rvdsetm:	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8) sendapar(8) rvdsend(8) rvdsend(8) rvdsetm(8) rvdsetm(8) rvdsend(8) rvdsetm(8) rvdsetm(8) rvdshow(8)
alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command rc.config: configuration file for startup disks using the IBM Small Computer System Interface /min, omin, fmin, m_in, mout, omout, fmout, m_out, badsect: create files to contain bad opendir, readdir, telldir, VI_Font, VI_GetFont, VI_DropFont: case: Academic Information Systems experimental display kill: sendapar: server. rvdsend - psp: planar ibm3812pp: IBM 3812 Pageprinter change logging level of Remote Virtual Disk (RVD) operations message from Remote Virtual Disk (RVD) set operations message on Remote Virtual Disk (RVD) show connections to Remote Virtual Disk (RVD)	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. scetdir, rewinddir, closedir: directory operations. select and manipulate fonts. select and manipulate fonts. selector in switch. self-tests. aedtest: IBM send a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD) sendapar: send APAR. serial port RS232C interface. server. server. rvdsedm: server. rvdsedm: server. rvdsedm: server. rvdsedm: server. rvdsedm: server. rvdsedm: server. rvdshow:	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8) sendapar(8) rvdsend(8) rvdsend(8) rvdsetm(8) rvdsetm(8) rvdsetm(8) rvdsetm(8) rvdsetm(8) rvdsehow(8) rvdshow(8) rvdshow(8) rvdshow(8) rvdshow(8) rvdshow(5)
alarm: ualarm: xcalc: X based xrefresh - refresh all windows on the setscreen: control display VI_Color: change curses: consoles: utility database of display onintr: process interrupts in command rc.config: configuration file for startup disks using the IBM Small Computer System Interface /min, omin, fmin, m_in, mout, omout, fmout, m_out, badsect: create files to contain bad opendir, readdir, telldir, VI_Font, VI_GetFont, VI_DropFont: case: Academic Information Systems experimental display kill: sendapar: server. rvdsend - psp: planar ibm3812pp: IBM 3812 Pageprinter change logging level of Remote Virtual Disk (RVD) operations message from Remote Virtual Disk (RVD) set operations message on Remote Virtual Disk (RVD) show connections to Remote Virtual Disk (RVD) force shutdown of Remote Virtual Disk (RVD)	scan a directory. scancode table. scandir, alphasort: scan a directory. scanf, fscanf, sscanf: formatted input conversion. schedule signal after specified time. schedule signal after specified time. scientific calculator. screen. screen access. screen color. screen functions with "optimal" cursor motion. screens. scripts. (SCSI) Adapter. sc: IBM 9332 scsiformat: format the IBM 9332 disk unit. sdiv, itom: multiple precision integer arithmetic. sectors. seekdir, rewinddir, closedir: directory operations. select and manipulate fonts. select or in switch. self-tests. aedtest: IBM send a signal to a process. send APAR. send control stream to Remote Virtual Disk (RVD) sendapar: send APAR. serial port RS232C interface. server. server. rvdchlog: server. rvdsed server. rvdsed server. rvdsetm: server. rvdsetm: server. rvdsetm:	scandir(3) keyboard_codes(5) scandir(3) scanf(3S) alarm(3C) ualarm(3) xcalc(1) xrefresh(1) setscreen(8) color(3G) curses(3X) consoles(5) csh(1) rc.config(5) sc(4) scsiformat(8c) mp(3X) badsect(8) directory(3) font(3G) csh(1) aedtest(8) kill(3F) sendapar(8) rvdsend(8) sendapar(8) rvdsend(8) rvdsend(8) rvdsetm(8) rvdsetm(8) rvdsetm(8) rvdsetm(8) rvdsetm(8) rvdsehow(8) rvdshow(8) rvdshow(8) rvdshow(8) rvdshow(8) rvdshow(5)

xrdb -	Corner Becourse Database Litility	vedb(1)
	Server Resource Database Utility	xrdb(1)
rvdlog: cause Remote Virtual Disk (RVD)		rvdlog(8) rvdhosts(8)
rvdhosts: print a list of RVD	servers	
logout: end	session.	csh(1)
stty, gtty:	set and get terminal state (defunct).	stty(3C)
*** ***	set: change value of shell variable	csh(1)
VI_Clip:	set clipping window	clip(3G)
utime:	set file times.	utime(3C)
pf:	set keyboard program-function keys	pf(1)
VI_Dash:	set line dash pattern	dash(3G)
VI_Width:	set line width.	width(3G)
VI_Merge:	set merge mode.	merge(3G)
server. rvdsetm:	set operations message on Remote Virtual Disk (RVD)	rvdsetm(8)
nice:	set program priority	nice(3C)
date: print and	set the date	date(1)
setuid, seteuid, setruid, setgid, setegid, setrgid:	set user and group ID	setuid(3)
seteny:	set variable in environment	csh(1)
a stream.	setbuf, setbuffer, setlinebuf: assign buffering to	setbuf(3S)
stream. setbuf,	setbuffer, setlinebuf: assign buffering to a	setbuf(3S)
setuid, seteuid, setruid, setgid,	setegid, setrgid: set user and group ID	setuid(3)
bottaid, bottaid, bottaid, bottaid,	setenv: set variable in environment.	csh(1)
user and group ID. setuid,	seteuid, setruid, setgid, setegid, setrgid: set	setuid(3)
	setfsent, endfsent: get file system descriptor file	getfsent(3)
entry. getfsent, getfsspec, getfsfile, getfstype,		
setuid, seteuid, setruid,	setgid, setegid, setrgid: set user and group ID	setuid(3)
getgrent, getgrgid, getgrnam,	setgrent, endgrent: get group file entry	getgrent(3)
gethostbyname, gethostbyaddr, gethostent,	sethostent, endhostent: get network host entry	gethostbyname(3N)
	setjmp, longjmp: non-local goto	setjmp(3)
crypt,	setkey, encrypt: DES encryption	crypt(3)
setbuf, setbuffer,	setlinebus: assign buffering to a stream	setbuf(3S)
syslog, openlog, closelog,	setlogmask: control system log.	syslog(3)
getnetent, getnetbyaddr, getnetbyname,	setnetent, endnetent: get network entry	getnetent(3N)
getprotoent, getprotobynumber, getprotobyname,	setprotoent, endprotoent: get protocol entry	getprotoent(3N)
entry. getpwent, getpwuid, getpwnam,	setpwent, endpwent, setpwfile: get password file	getpwent(3)
getpwent, getpwuid, getpwnam, setpwent, endpwent,	setpwfile: get password file entry	getpwent(3)
setuid, seteuid, setruid, setgid, setegid,	setrgid: set user and group ID	setuid(3)
group ID. setuid, seteuid,	setruid, setgid, setegid, setrgid: set user and	setuid(3)
	setscreen: control display screen access	setscreen(8)
getservent, getservbyport, getservbyname,	setservent, endservent: get service entry	getservent(3N)
for changing/ random, srandom, initstate,	setstate: better random number generator; routines	random(3)
xsetroot: X window system root window parameter	setting utility.	xsetroot(1)
getttyent, getttynam,	setttyent, endttyent: get ttys file entry.	getttyent(3)
set user and group ID.	setuid, seteuid, setruid, setgid, setegid, setrgid:	setuid(3)
xset - X window system user	setup program	xset(1)
getusershell,	setusershell, endusershell: get legal user shells	getusershell(3)
exit: leave	shell	csh(1)
system: issue a	shell command.	system(3)
eval: re-evaluate	shell data	csh(1)
popd: pop	shell directory stack	csh(1)
pushd: push	shell directory stack	csh(1)
alias:	shell macros.	csh(1)
suspend: suspend a	shell, resuming its superior.	csh(1)
set: change value of	shell variable.	csh(1)
@: arithmetic on	shell variables.	csh(1)
unset: discard	shell variables.	csh(1)
exec: overlay	shell with specified command.	csh(1)
setusershell, endusershell: get legal user	shells, getusershell,	getusershell(3)
secuser shen, endusershen. get legal user	shift: manipulate argument list.	csh(1)
long	short: integer object conversion.	long(3F)
long, server. rvdshow:	show connections to Remote Virtual Disk (RVD)	rvdshow(8)
rvdshut: force		, ,
	shutdown of Remote Virtual Disk (RVD) server	rvdshut(8)
calls.	siginterrupt: allow signals to interrupt system	siginterrupt(3)
pause: stop until	signal	pause(3C)
signal: change the action for a	signal.	signal(3F)
alarm: schedule	signal after specified time.	alarm(3C)
ualarm: schedule	signal after specified time.	ualarm(3)
	signal: change the action for a signal	signal(3F)
sigvec: software	signal facilities.	sigvec(2)
signal: simplified software	signal facilities.	signal(3C)
psignal, sys_siglist: system	signal messages	psignal(3)
· · · · · · · · · · · · · · · · · · ·	signal: simplified software signal facilities	signal(3C)
kill: send a	signal to a process	kill(3F)
(temporary). infnan:	signals invalid floating-point operations on a VAX	infnan(3M)
siginterrupt: allow	signals to interrupt system calls	siginterrupt(3)
5 F		
alamal.	sigvec: software signal facilities	sigvec(2)
signal:		signal(3C)
trigonometric functions and their inverses.	simplified software signal facilities	

	sinh, cosh, tanh: hyperbolic functions	sinh(3M)
diskpart: calculate default disk partition	sizes	diskpart(8)
	sleep: suspend execution for interval	sleep(3F) sleep(3)
sc: IBM 9332 disks using the IBM	Small Computer System Interface (SCSI) Adapter	sc(4)
spline: interpolate	smooth curve.	spline(1G) sigvec(2)
sigvec: signal: simplified	software signal facilities	signal(3C)
support: hardware and	software support information.	support(1)
qsort: quicker	sort	qsort(3)
qsort: quick	sort	qsort(3F) csh(1)
line, circle, arc, move, cont, point, linemod,	space, closepl: graphics interface. /erase, label,	
	speaker: console speaker interface	speaker(4)
speaker: console exec: overlay shell with	speaker interface	speaker(4) csh(1)
truncate, ftruncate: truncate a file to a	specified length.	truncate(2)
alarm: schedule signal after	specified time.	alarm(3C)
alarm: execute a subroutine after a	specified time.	alarm(3F)
ualarm: schedule signal after vdabort: abort and	specified time	ualarm(3) vdabort(8)
vdspin, vdspind: spin up or	spin down a Remote Virtual Disk (RVD)	vdspin(2)
vdspin, vdspind:	spin up or spin down a Remote Virtual Disk (RVD)	vdspin(2)
spinup, spindown: rvdfiush:	spin up/down Remote Virtual Disk (RVD) pack spindown client's Remote Virtual Disk (RVD) packs	spinup(8) rvdflush(8)
rvddown: force	spindown of a Remote Virtual Disk (RVD) pack	rvddown(8)
pack. spinup,	spindown: spin up/down Remote Virtual Disk (RVD) .	spinup(8)
(RVD) pack.	spinup, spindown: spin up/down Remote Virtual Disk spline: interpolate smooth curve	spinup(8) spline(1G)
frexp, Idexp, modf:	split into mantissa and exponent.	frexp(3)
Pageprinter. ppt:	spooling system filter for the IBM 3812	
printf, fprintf,	sprintf: formatted output conversion	printf(3S)
cbrt, cbrt, sqrt: cube root,	squre root	sqrt(3M) sqrt(3M)
rand,	srand: random number generator.	rand(3C)
generator; routines for changing/ random,	srandom, initstate, setstate: better random number	random(3)
scanf, fscanf,	sscanf: formatted input conversion. st: streaming-tape interface	scanf(3S) st(4)
popd: pop shell directory	stack	csh(1)
pushd: push shell directory	stack	csh(1)
sautil: stdio:	standalone utility package standard buffered input/output package	sautil(8r) stdio(3S)
stdio. stdemul:	standard output emulator	stdemul(4)
rc.config: configuration file for	startup scripts	rc.config(5)
raturn machine and process floating point	stat, Istat, Istat: get file status	stat(3F) getfloatstate(2)
return machine and process floating point stty, gtty: set and get terminal	state (defunct).	
if: conditional	statement	csh(1)
tailor: work	station customizing assistance	
hashstat: print command hashing cause Remote Virtual Disk (RVD) server to log	statistics. rvdlog:	csh(1) rvdlog(8)
vdstats: acquire client Remote Virtual Disk (RVD)	statistics.	vdstats(2)
vdstats: list client Remote Virtual Disk (RVD)	statistics	vdstats(8)
vmstat: report virtual memory exit: terminate process with	statistics	vmstat(1) exit(3F)
stat, Istat, fstat: get file	status.	stat(3F)
printer3812: IBM 3812 Pageprinter	status information	printer3812(5)
ferror, feof, clearerr, fileno: stream	status inquiries	ferror(3S) stdemul(4)
	stdio: standard buffered input/output package	stdio(3S)
	stop: halt a job or process	csh(1)
halt: pause:	stop the processor	halt(8) pause(3C)
subroutines. dbminit, fetch,	store, delete, firstkey, nextkey: data base	dbm(3X)
strlen, index, rindex: string operations.	streat, strneat, stremp, strnemp, strepy, strnepy,	string(3)
rindex: string operations. strcat, strncat, operations. strcat, strncat, strcmp, strncmp,	strcmp, strncmp, strcpy, strncpy, strlen, index, strcpy, strncpy, strlen, index, rindex: string	string(3) string(3)
fclose, fflush: close or flush a	stream	fclose(3S)
fopen, freopen, fdopen: open a	stream	fopen(3S)
fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from	stream. getc,	fseek(3S)
gets, fgets: get a string from a	stream	getc(3S) gets(3S)
putchar, fputc, putw: put character or word on a	stream. putc,	putc(3S)
puts, fputs: put a string on a	stream	puts(3S)
setbuffer, setlinebuf: assign buffering to a ungete: push character back into input	stream. setbuf,	setbuf(3S) ungetc(3S)
O Paris same and man made		

Communication Control of Control	and a second sec	C(2C)
ferror, feof, clearerr, fileno:	stream status inquiries	ferror(3S)
rcmd, rresvport, ruserok: routines for returning a rexec: return	stream to a remote command	` '
rydsend - send control	stream to Remote Virtual Disk (RVD) server	rvdsend(8)
tbuffer:	streaming tape buffered read	tbuffer(8)
st:	streaming-tape interface.	st(4)
fdate: return date and time in an ASCII	string.	fdate(3F)
VI String: draw a	string	string(3G)
gets, fgets: get a	string from a stream	gets(3S)
puts, fputs: put a	string on a stream	
bcopy, bcmp, bzero, ffs: bit and byte	string operations	bstring(3)
strncmp, strcpy, strncpy, strlen, index, rindex:	string operations. strcat, strncat, strcmp,	string(3)
streat, strncat, stremp, strncmp, strepy, strncpy,	strlen, index, rindex: string operations	string(3)
index, rindex: string operations. strcat,	strncat, strcmp, strncmp, strcpy, strncpy, strlen,	string(3)
string operations. streat, strncat, stremp,	strncmp, strcpy, strncpy, strlen, index, rindex:	
streat, strneat, stremp, strnemp, strepy,	strncpy, strlen, index, rindex: string operations	
font3812: font	structures for 3812 fonts	
	stty, gtty: set and get terminal state (defunct)	stty(3C)
alarm: execute a	subroutine after a specified time	alarm(3F)
VI_Login, VI_Logout: begin logging	subroutine calls and close a log file	log(3G)
VI_Init, VI_Term: initialize and terminate the	subroutine interface	init(3G)
fetch, store, delete, firstkey, nextkey: data base	subroutines. dbminit,	dbm(3X)
intro: introduction to display graphics	subroutines	
dbm_nextkey, dbm_error, dbm_clearerr: data base	subroutines. /dbm_store, dbm_delete, dbm_firstkey,	ndbm(3)
lib2648:	subroutines for the HP 2648 graphics terminal	lib2648(3X)
xwininfo - X Window System window information	summarizer	xwininfo(1)
suspend: suspend a shell, resuming its	superior	csh(1)
intro: introduction to special files and hardware	support	intro(4)
	support: hardware and software support information	support(1)
support: hardware and software	support information.	support(1)
suspend:	suspend a shell, resuming its superior	csh(1)
sleep:	suspend execution for an interval	sleep(3F)
sleep:	suspend execution for interval.	sleep(3)
usleep:	suspend execution for interval.	usleep(3)
	suspend: suspend a shell, resuming its superior	csh(1)
	swab: swap bytes.	swab(3)
swab:	swap bytes.	swab(3)
nextdouble, nextfloat, fptestround, fpsetround,	swapround, fptestflag, fpsetflag, /infinity,	ieee(3)
breaksw: exit from	switch.	csh(1)
case: selector in	switch.	csh(1)
default: catchall clause in	switch.	csh(1)
endsw: terminate	switch.	csh(1)
	switch: multi-way command branch	csh(1)
cvtsym: convert	symbol table.	cvtsym(8)
makesym: make debugger	symbol table	makesym(8) dbx(5)
dbx: dbx	symbol table information	
narrar	system: system can interface program	
perror, system log.	syslog, openlog, closelog, setlogmask: control	
system log.	syslogd: log systems messages	syslogd(8)
perror, sys errlist,	sys_nerr: system error messages.	perror(3)
perfor, sys_crimst, psignal,	sys siglist: system signal messages.	
interfaces for the IBM Academic Information	Systems experimental display. aedemul: graphics	aedemul(4)
ibmaed, aed: IBM Academic Information	Systems experimental display interface	ibmaed(4)
aedtest: IBM Academic Information	Systems experimental display self-tests	aedtest(8)
syslogd: log	systems messages.	syslogd(8)
rehash: recompute command hash	table.	csh(1)
unhash: discard command hash	table	csh(1)
cvtsym: convert symbol	table	cvtsym(8)
keyboard codes: keyboard scancode	table	keyboard_codes(5)
makesym: make debugger symbol	table	makesym(8)
Remote Virtual Disk (RVD) server configuration	table. rvddb:	rvddb(5)
dbx: dbx symbol	table information.	dbx(5)
fdisk: boot record partition	table maintenance utility.	fdisk(8)
width3812: build width	tables for IBM 3812 Pageprinter fonts	width3812(8)
	tailor: work station customizing assistance	tailor(8)
functions and their inverses. sin, cos,	tan, asin, acos, atan, atan2: trigonometric	sin(3M)
sinh, cosh,	tanh: hyperbolic functions.	sinh(3M)
restore Remote Virtual Disk (RVD) packs to and from	tape. savervd, zaprvd, savephys: back up and	savervd(8)
tar:	tape archiver.	tar(1)
tologo trond twrite travely taking tetata: 677	tape buffered read	tbuffer(8)
tclose, tread, twrite, trewin, tskipf, tstate: f77	tape I/O. topen,	topen(3F)
mt: magnetic	tape manipulating program	mt(1)
restore.tape, restore.net: install system from	tape or over network	restore.tape(8)
network, restore.	tape, restore net: install system from tape or over tar: tape archiver	restore.tape(8) tar(1)
	tall tape aromyer.	·ai(1)

			2.040
	tb: line discipline for digitizing devices		tb(4)
	tbuffer: streaming tape buffered read		tbuffer(8)
tape I/O. topen,	tclose, tread, twrite, trewin, tskipf, tstate: f77		topen(3F)
index, rindex, Inblnk, len:	tell about character objects		index(3F)
operations. opendir, readdir,	telldir, seekdir, rewinddir, closedir: directory		directory(3)
signals invalid floating-point operations on a VAX	(temporary). infnan:		infnan(3M)
ptfinstall: install a Program	Temporary Fix (PTF)		ptfinstall(8)
lib2648: subroutines for the HP 2648 graphics	terminal.		lib2648(3X)
ttyname, isatty, ttyslot: find name of a	terminal		ttyname(3)
xterm: X window system	terminal emulator		xterm(1)
Xtty: routines to provide	terminal emulator windows		xtty(3X)
tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs:	terminal independent operation routines		termcap(3X
tty: general	terminal interface		tty(4)
ttynam, isatty: find name of a	terminal port		ttynam(3F)
stty, gtty: set and get	terminal state (defunct).		stty(3C)
			* * * * * * * * * * * * * * * * * * * *
wait: wait for a process to	terminate.		wait(3F)
output. exit:	terminate a process after flushing any pending	• •	exit(3)
abort:	terminate abruptly with memory image		abort(3F)
endif:	terminate conditional		csh(1)
end:	terminate loop.		csh(1)
exit:	terminate process with status		exit(3F)
endsw:	terminate switch.		csh(1)
	terminate the subroutine interface.		
VI_Init, VI_Term: initialize and			init(3G)
Accelerator. afpacode: load,	, ,	• •	afpacode(8r)
pprint: print	text files on IBM 3812 Pageprinter		pprint(1)
Xtext: routines to provide simple	text output windows		xtext(3X)
terminal independent operation routines.	tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs:		termcap(3X)
independent operation routines. tgetent, tgetnum,	tgetflag, tgetstr, tgoto, tputs: terminal		termcap(3X
independent operation routines, tgetent,	tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal	• •	termcap(3X
		• •	
operation routines. tgetent, tgetnum, tgetflag,	tgetstr, tgoto, tputs: terminal independent		termcap(3X)
routines. tgetent, tgetnum, tgetflag, tgetstr,	tgoto, tputs: terminal independent operation		termcap(3X)
acos, atan, atan2: trigonometric functions and	their inverses. sin, cos, tan, asin,		sin(3M)
VI Tile:	tile a rectangle		tile(3G)
alarm: schedule signal after specified	time		alarm(3C)
alarm: execute a subroutine after a specified	time.		alarm(3F)
etime, dtime: return elapsed execution	time.		` '
			etime(3F)
time, ftime: get date and	time		time(3C)
time, ctime, ltime, gmtime: return system	time	• •	time(3F)
ualarm: schedule signal after specified	time		ualarm(3)
time:	time command		csh(1)
	time, ctime, ltime, gmtime: return system time		time(3F)
e de la companya de	time, ftime: get date and time.		time(3C)
fdate: return date and	time in an ASCII string.		fdate(3F)
idate, itime: return date or	time in numerical form.		idate(3F)
	time: time command		csh(1)
gmtime, asctime, timezone: convert date and	time to ASCII. ctime, localtime,		ctime(3)
times: get process	times		times(3C)
utime: set file	times.		utime(3C)
	times: get process times.		times(3C)
atima localtima amtima acatima	timezone: convert date and time to ASCII		ctime(3)
ctime, localtime, gmtime, asctime,		• •	` '
	tn3270: full-screen remote login to IBM VM/CMS.	• •	tn3270(1)
isgraph, iscntrl, isascii, toupper, tolower,	toascii: character classification macros. /isprint,		ctype(3)
popen, pclose: initiate I/O	to/from a process		popen(3)
lan: IBM RT PC	Token-Ring Adapter		lan(4)
landump: dump IBM	Token-Ring Personal Computer Adapter		landump(8r)
/isprint, isgraph, iscntrl, isascii, toupper,	tolower, toascii: character classification macros		ctype(3)
			* * * * * * * * * * * * * * * * * * * *
tstate: f77 tape I/O.	topen, tclose, tread, twrite, trewin, tskipf,		topen(3F)
/ispunct, isprint, isgraph, iscntrl, isascii,	toupper, tolower, toascii: character classification/		ctype(3)
tgetent, tgetnum, tgetflag, tgetstr, tgoto,	tputs: terminal independent operation routines		termcap(3X)
ptrace: process	trace		ptrace(2)
goto: command	transfer		csh(1)
trpfpe, fpecnt:	trap and repair floating point faults		trpfpe(3F)
trapov:	trap and repair floating point overflow		trapov(3F)
traper:	trap arithmetic errors.		
traper:			traper(3F)
	traper: trap arithmetic errors.		traper(3F)
	trapov: trap and repair floating point overflow		trapov(3F)
I/O. topen, tclose,	tread, twrite, trewin, tskipf, tstate: f77 tape		topen(3F)
topen, tclose, tread, twrite,	trewin, tskipf, tstate: f77 tape I/O		topen(3F)
sin, cos, tan, asin, acos, atan, atan2:	trigonometric functions and their inverses		sin(3M)
ptroff: print			ptroff(1)
piron. pine	troff preprocessor for drawing simple pictures.		pic(1)
faults.	trpfpe, fpecnt: trap and repair floating point		trpfpe(3F)
truncate, ftruncate:	truncate a file to a specified length	• •	truncate(2)
length.	truncate, struncate: truncate a file to a specified		truncate(2)
topen, tclose, tread, twrite, trewin,	tskipf, tstate: f77 tape I/O		topen(3F)
tonen tolose tread twrite trewin tskinf			topen(3E)

	tty: general terminal interface	tty(4)
	ttynam, isatty: find name of a terminal port	ttynam(3F)
	ttyname, isatty, ttyslot: find name of a terminal	ttyname(3)
getttyent, getttynam, setttyent, endttyent: get	ttys file entry.	getttyent(3)
ttyname, isatty,	ttyslot: find name of a terminal	ttyname(3)
topen, tclose, tread,	twrite, trewin, tskipf, tstate: f77 tape I/O	topen(3F)
	ualarm: schedule signal after specified time	ualarm(3)
getpw: get name from	uid	getpw(3C)
	umask: change or display file creation mask	csh(1)
Ethernet.	un: IBM RT PC Baseband Adapter for use with	un(4)
	unalias: remove aliases	csh(1)
xwud - X Window System, window image	undumper	xwud(1)
	ungetc: push character back into input stream	ungetc(3S)
	unhash: discard command hash table	csh(1)
mktemp: make a	unique file name	mktemp(3)
scsiformat: format the IBM 9332 disk	unit	scsiformat(8c)
flush: flush output to a logical	unit	flush(3F)
fseek, ftell: reposition a file on a logical	unit	fseek(3F)
getc, fgetc: get a character from a logical	unit	getc(3F)
putc, fputc: write a character to a fortran logical	unit	putc(3F)
learn: computer aided instruction about	UNIX	learn(1)
system: execute a	UNIX command	system(3F)
•,•••••	unlimit: remove resource limitiations	csh(1)
	unlink: remove a directory entry.	unlink(3F)
/logb, scalb, rint, classdouble, classfloat, isnan,	unordered, finite, infinity, nextdouble, nextfloat,/	ieee(3)
(1000) seems, this, simulation, simulation, islinit,	unset: discard shell variables.	csh(1)
	unsetenv: remove environment variables	csh(1)
and from tape. savervd, zaprvd, savephys: back	up and restore Remote Virtual Disk (RVD) packs to .	savervd(8)
utilities.	up, down: client Remote Virtual Disk (RVD)	up(1)
		* * * * * * * * * * * * * * * * * * * *
vdspin, vdspind: spin	up or spin down a Remote Virtual Disk (RVD)	vdspin(2)
spinup, spindown: spin	up/down Remote Virtual Disk (RVD) pack	spinup(8)
un: IBM RT PC Baseband Adapter for	use with Ethernet.	un(4)
cvt00to12: convert IBM 3820 and IBM 3800 fonts for	use with the IBM 3812 Pageprinter. /cvt20to12,	cvt3812(8)
login: login new	user.	csh(1)
seteuid, setruid, setgid, setegid, setrgid: set	user and group ID. setuid,	setuid(3)
nfcomment: a	user interface to the notesfile system	nfcomment(3)
getuid, getgid: get	user or group ID of the caller	getuid(3F)
xset - X window system	user setup program	xset(1)
getusershell, setusershell, endusershell: get legal	user shells	getusershell(3)
getlog: get	user's login name.	getlog(3F)
(SCSI) Adapter. sc: IBM 9332 disks	using the IBM Small Computer System Interface	sc(4)
• • •	usleep: suspend execution for interval	usleep(3)
up, down: client Remote Virtual Disk (RVD)	utilities	up(1)
modmap, xprkbd - X Window System keyboard modifier	utilities	xmodmap(1)
fdisk: boot record partition table maintenance	utility	fdisk(8)
minidisk: minidisk maintenance	utility	minidisk(8r)
xrdb - Server Resource Database	Utility	xrdb(1)
X window system root window parameter setting	utility. xsetroot:	xsetroot(1)
consoles:	utility database of display screens	consoles(5)
sautil: standalone	utility package	sautil(8r)
vtimes: get information about resource	utilization.	vtimes(3C)
8	utime: set file times.	utime(3C)
.)		uwm(1)
	valloc: aligned memory allocator.	valloc(3C)
abs: integer absolute	value	abs(3)
hypot, cabs: Euclidean distance, complex absolute	•	hypot(3M)
fabs, floor, ceil: absolute		floor(3M)
getenv:	value for environment name	getenv(3)
getenv: get		getenv(3F)
set: change	value of shell variable	csh(1)
dfimin, dfimax, dffrac, inmax: return extreme	values. filmin, filmax, ffrac,	flmin(3F)
rand, drand, irand: return random	values.	rand(3F)
htonl, htons, ntohl, ntohs: convert	values between host and network byte order	byteorder(3N
	varargs: variable argument list.	varargs(3)
set: change value of shell	variable	csh(1)
varargs:	variable argument list.	varargs(3)
setenv: set	variable in environment	csh(1)
@: arithmetic on shell	variables	csh(1)
unset: discard shell	variables	csh(1)
unsetenv: remove environment	variables	csh(1)
getenv: get value of environment	variables	getenv(3F)
signals invalid floating-point operations on a	VAX (temporary). infnan:	infnan(3M)
· · · · · · · · · · · · · · · · ·	vdabort: abort and spin down a drive	vdabort(8)
	vddb: Remote Virtual Disk (RVD) data base manager	vddb(8)
Virtual Disk (RVD).	vdspin, vdspind: spin up or spin down a Remote	vdspin(2)
(RVD). vdspin,	vdspind: spin up or spin down a Remote Virtual Disk	vdspin(2)
/ /- · #-F		4

to the second		
statistics.	vdstats: acquire client Remote Virtual Disk (RVD)	vdstats(2)
statistics.	vdstats: list client Remote Virtual Disk (RVD) verification	vdstats(8) assert(3)
assert: program 3812 Pageprinter.	verification	vgrind(1)
Joiz i agepinter.	VI ALine, VI RLine: draw a line	line(3G)
	VI AMove, VI RMove: move the current point	move(3G)
	VI Circle: draw a circle	circle(3G)
	VI_Clip: set clipping window	clip(3G)
	VI_Color: change screen color	color(3G)
	VI_Copy: copy an area.	copy(3G)
VI MD-f-G VI ED-f-G VI E-G	VI_Dash: set line dash pattern.	dash(3G)
VI_MDefnCur, VI_FDefnCur, VI_EnCur, VI_Font, VI_GetFont,	VI_DisCur, VI_PosnCur: control the display cursor VI_DropFont: select and manipulate fonts	cursor(3G)
display cursor. VI_MDefnCur, VI_FDefnCur,	VI_DropFont: select and manipulate fonts	font(3G) cursor(3G)
control the display cursor. VI_MDefnCur,	VI FDefnCur, VI EnCur, VI DisCur, VI PosnCur:	cursor(3G)
VI MImage,	VI FImage: draw an image	image(3G)
manipulate fonts.	VI_Font, VI_GetFont, VI_DropFont: select and	font(3G)
-	VI_Force: force output of graphics orders	force(3G)
VI_MRead,	VI_FRead: read display data	read(3G)
fonts. VI_Font,	VI_GetFont, VI_DropFont: select and manipulate	font(3G)
subroutine interface.	VI_Init, VI_Term: initialize and terminate the	init(3G)
and close a log file.	VI_Login, VI_Logout: begin logging subroutine calls	log(3G)
a log file. VI_Login,	VI_Logout: begin logging subroutine calls and close .	log(3G)
VI_PosnCur: control the display cursor.	VI_MDefnCur, VI_FDefnCur, VI_EnCur, VI_DisCur,	cursor(3G)
	VI_Merge: set merge mode	merge(3G) image(3G)
	VI MRead, VI FRead: read display data	read(3G)
VI MDefnCur, VI FDefnCur, VI EnCur, VI DisCur,	VI PosnCur: control the display cursor	cursor(3G)
VI QPoint, VI QWidth: query graphics parameters.	VI QClip, VI QColor, VI QDash, VI QFont, VI QMerge	
VI_QPoint, VI_QWidth: query graphics/ VI_QClip,	VI_QColor, VI_QDash, VI_QFont, VI_QMerge,	query(3G)
VI_QWidth: query graphics/ VI_QClip, VI_QColor,	VI_QDash, VI_QFont, VI_QMerge, VI_QPoint,	query(3G)
graphics/ VI_QClip, VI_QColor, VI_QDash,	VI_QFont, VI_QMerge, VI_QPoint, VI_QWidth: query	query(3G)
VI_QClip, VI_QColor, VI_QDash, VI_QFont,	VI_QMerge, VI_QPoint, VI_QWidth: query graphics/	query(3G)
VI_QClip, VI_QColor, VI_QDash, VI_QFont, VI_QMerge,	VI_QPoint, VI_QWidth: query graphics parameters	query(3G)
VI_QDash, VI_QFont, VI_QMerge, VI_QPoint, VI_ALine,	VI_QWidth: query graphics parameters. /VI_QColor,	query(3G) line(3G)
VI Abne,	VI RMove: move the current point.	move(3G)
rvd: Remote	Virtual Disk protocol.	rvd(4p)
newvd: create a new filesystem on a Remote	Virtual Disk (RVD)	newvd(8)
vdspin, vdspind: spin up or spin down a Remote	Virtual Disk (RVD)	vdspin(2)
vddb: Remote	Virtual Disk (RVD) data base manager	vddb(8)
rvddown: force spindown of a Remote	Virtual Disk (RVD) pack	rvddown(8)
spinup, spindown: spin up/down Remote rvdexch: exchange names of two Remote	Virtual Disk (RVD) pack	spinup(8) rvdexch(8)
rydflush: spindown client's Remote	Virtual Disk (RVD) packs	rvdflush(8)
/zaprvd, savephys: back up and restore Remote	Virtual Disk (RVD) packs to and from tape	savervd(8)
rvdchlog: change logging level of Remote	Virtual Disk (RVD) server	rvdchlog(8)
rvdgetm: get operations message from Remote	Virtual Disk (RVD) server	rvdgetm(8)
rvdsend - send control stream to Remote	Virtual Disk (RVD) server	rvdsend(8)
rvdsetm: set operations message on Remote	Virtual Disk (RVD) server	rvdsetm(8)
rydshow: show connections to Remote	Virtual Disk (RVD) server	rvdshow(8)
rvdshut: force shutdown of Remote	Virtual Disk (RVD) server	rvdshut(8) rvddb(5)
rvddb: Remote rvdsrv: Remote	Virtual Disk (RVD) server configuration table Virtual Disk (RVD) server daemon	rvdao(3) rvdsrv(8)
rydlog: cause Remote	Virtual Disk (RVD) server daemon	rvdlog(8)
vdstats: acquire client Remote	Virtual Disk (RVD) statistics	vdstats(2)
vdstats: list client Remote	Virtual Disk (RVD) statistics	vdstats(8)
up, down: client Remote	Virtual Disk (RVD) utilities	up(1)
/etc/rvd/rvdtab: information about client Remote	Virtual Disks (RVDs)	rvdtab(5)
vmstat: report	virtual memory statistics.	vmstat(1)
	VI_Run: process a log file.	run(3G)
interface VI Init	VI_String: draw a string	string(3G)
interface. VI_Init,	VI_Term: initialize and terminate the subroutine VI_Tile: tile a rectangle	init(3G) tile(3G)
	VI Width: set line width	width(3G)
consumption.	vlimit: control maximum system resource	vlimit(3C)
tn3270: full-screen remote login to IBM	VM/CMS	tn3270(1)
	vmstat: report virtual memory statistics.	vmstat(1)
	vtimes: get information about resource utilization	vtimes(3C)
wait:	wait for a process to terminate.	wait(3F)
wait:	wait for background processes to complete wait: wait for a process to terminate	csh(1) wait(3F)
	wait: wait for background processes to complete	csh(1)
crash:	what happens when the system crashes	crash(8r)
crash:	what happens when the system crashes	crash(8V)
crash: what happens	when the system crashes	crash(8r)

crash: what happens	when the system crashes	
1 1 2	while: repeat commands conditionally	csh(1)
break: exit	while/foreach loop	csh(1)
VI_Width: set line	width.	width(3G)
width3812: build Pageprinter fonts.	width tables for IBM 3812 Pageprinter fonts	width3812(8)
VI Clip: set clipping	width3812: build width tables for IBM 3812 window.	width3812(8) clip(3G)
xpr: print X	window dump.	xpr(1)
xwd - X Window System,	window image dumper	xwd(1)
xwud - X Window System,	window image undumper	xwud(1)
xwininfo - X Window System	window information summarizer	xwininfo(1)
xmh: X	window interface to the mh Mail Handler	xmh(1)
wm: a simple real-estate-driven	window manager	wm(1)
uwm -	Window Manager Client Application of X	uwm(1)
xsetroot: X window system root	window parameter setting utility.	xsetroot(1)
bitmap: bitmap editor for X	window system.	bitmap(1)
xhost - X	window system access control program	xhost(1)
xclock - X	Window System, analog / digital clock	xclock(1)
xfd - X	window system font displayer	xfd(1)
xlsfonts - X	window system font list displayer	xlsfonts(1)
xinit - X	window system initializer	xinit(1)
Xlib: C Language X	Window System Interface Library	Xlib(3X)
xmodmap, xprkbd - X	Window System keyboard modifier utilities	xmodmap(1)
xload - X	window system load average display	xload(1)
xprop - X	Window System property displayer	xprop(1)
utility. xsetroot: X	window system root window parameter setting ,	xsetroot(1)
xterm: X	window system terminal emulator	xterm(1)
xset - X	window system user setup program	xset(1)
xwd - X	Window System, window image dumper	xwd(1)
xwud - X	Window System, window image undumper	xwud(1)
xwininfo - X	Window System window information summarizer	xwininfo(1)
Xtext: routines to provide simple text output	windows.	xtext(3X)
Xtty: routines to provide terminal emulator	windows	xtty(3X)
xrefresh - refresh all	windows on the screen.	xrefresh(1)
	wm: a simple real-estate-driven window manager	wm(1)
getc, getchar, fgetc, getw: get character or	word from stream	getc(3S)
putc, putchar, fputc, putw: put character or	word on a stream.	putc(3S)
tailor:	work station customizing assistance	tailor(8)
getcwd: get pathname of current	working directory	getcwd(3F)
getwd: get current	working directory pathname.	getwd(3)
putc, fputc:	write a character to a fortran logical unit	putc(3F) dosread(1)
dosread: read, xcalc:	write, dir, delete on PC-DOS diskette	
XMenu -	X Deck of cards Menu System.	xcalc(1) XMenu(3X)
events. xemul:	X input emulator for queuing keyboard and mouse	xemul(4)
uwm - Window Manager Client Application of	X : 4	uwm(1)
xpr: print	X window dump.	xpr(1)
xmh:	X window interface to the mh Mail Handler	xmh(1)
	X window system.	
	X window system access control program	xhost(1)
	X Window System, analog / digital clock	xclock(1)
xfd -	X window system font displayer	xfd(1)
xlsfonts -	X window system font list displayer	xlsfonts(1)
xinit -	X window system initializer	xinit(1)
Xlib: C Language	X Window System Interface Library	Xlib(3X)
xmodmap, xprkbd -	X Window System keyboard modifier utilities	xmodmap(1)
xload -	X window system load average display	xload(1)
xprop -	X Window System property displayer	xprop(1)
utility. xsetroot:	X window system root window parameter setting	xsetroot(1)
xterm:	X window system terminal emulator	xterm(1)
xset -	X window system user setup program	xset(1)
xwd -	X Window System, window image dumper	xwd(1)
xwud -	X Window System, window image undumper	xwud(1)
xwininfo -	X Window System window information summarizer	xwininfo(1)
	xcalc: X based scientific calculator.	xcalc(1)
	xclock - X Window System, analog / digital clock	xclock(1)
mouse events.	xemul: X input emulator for queuing keyboard and	xemul(4)
ns_addr, ns_ntoa:	Xerox NS(tm) address conversion routines	ns(3N)
	xfd - X window system font displayer	xfd(1)
	xhost - X window system access control program	xhost(1)
	xinit - X window system initializer	xinit(1)
	xload - X window system load average display	Xlib(3X) xload(1)
	xisfonts - X window system font list displayer	xisfonts(1)
	XMenu - X Deck of cards Menu System	XMenu(3X)
utilities.	xmodmap, xprkbd - X Window System keyboard modifier	xmodmap(1)

xor, not, rshift, Ishift bitwise functions bit(3F)
xpr: print X window dump xpr(1)
xprkbd - X Window System keyboard modifier xmodmap(1)
xprop - X Window System property displayer xprop(1)
xrdb - Server Resource Database Utility xrdb(1)
xrefresh - refresh all windows on the screen xrefresh(1)
xset - X window system user setup program xset(1)
xsetroot: X window system root window parameter xsetroot(1)
xterm: X window system terminal emulator xterm(1)
Xtext: routines to provide simple text output $\dots x$ text(3X)
Xtty: routines to provide terminal emulator xtty(3X)
xwd - X Window System, window image dumper xwd(1)
xwininfo - X Window System window information xwininfo(1)
xwud - X Window System, window image undumper xwud(1)
y0, y1, yn: Bessel functions j0(3M)
y1, yn: Bessel functions j0(3M)
yn: Bessel functions j0(3M)
zaprvd, savephys: back up and restore Remote savervd(8)

Section 1. Commands and Application Programs

This section describes publicly-accessible commands of general utility. Man pages found in IBM/4.3, but not in 4.3BSD, are marked with an asterisk (*). Man pages marked with a section symbol (§) are at a level earlier than 4.3BSD.

- adb§
- aedjournal*
- aedrunner*
- andrew*
- as§
- bitprt*
- cc
- colpro*
- date
- dbx
- dosread*
- dumpaed*
- dumpapa16*
- dumpapa8*
- dumpapa8c*
- error
- f77§
- hc*
- kbdlock*

- ld§
- mset
- mt
- pcc§
- pf*
- pic*
- pp*
- pprint*
- prfl*
- proff*
- ptroff*
- scale*
- support*
- tar
- tn3270
- up*
- vgrind
- vmstat
- Xibm*
- xwindows*

NAME

adb - debugger

SYNOPSIS

$$adb [-w] [-k] [-Idir] [objfil [corfil]]$$

DESCRIPTION

Adb is a general-purpose debugging program. It may be used to examine files and to provide a controlled environment for the execution of programs.

Objfil is normally an executable program file, preferably containing a symbol table; if not, the symbolic features of adb cannot be used, although the file can still be examined. The default for objfil is a.out. Corfil is assumed to be a core image file produced after executing objfil; the default for corfil is core.

Requests to adb are read from the standard input and responses are to the standard output. If the —w flag is present, both objfil and corfil are created, if needed, and opened for reading and writing so that files can be modified using adb.

The $-\mathbf{k}$ option makes adb do kernel memory mapping; it should be used when corefil is a crash dump or |dev|mem.

The -I option specifies a directory where files to be read with \$ or \$ < are sought; the default is |usr|lib/adb.

Adb ignores QUIT; INTERRUPT causes adb to stop processing the current command and await a new command.

In general, requests to adb are of the form:

If address is present, dot is set to address. Initially, dot is set to 0. For most commands, count specifies how many times the command executes. The default count is 1. Address and count are expressions. The interpretation of an address depends on the context in which it is used. If a subprocess is being debugged, addresses are interpreted in the usual way in the address space of the subprocess. If the operating system is being debugged -- either post-mortem or using the special file |dev|mem to examine or modify memory interactively -- the maps are set to map the kernel virtual addresses which start at 0xe00000000 on the IBM RT PC.

EXPRESSIONS

- The value of dot.
- + The value of *dot* incremented by the current increment.
- ^ The value of *dot* decremented by the current increment.
- " The last address typed.

integer A number. The prefixes 00 and 00 ("zero oh") force interpretation in octal radix; the prefixes 0t and 0T force interpretation in decimal radix; the prefixes 0x and 0X force interpretation in hexadecimal radix. Thus 0020 = 0t16 = 0x10 = sixteen. If no prefix appears, the default radix is used (see the \$d command). The default radix is initially hexadecimal. The hexadecimal digits are 0123456789abcdefABCDEF with the obvious values. Note that a hexadecimal number whose leading digit is alphabetic must have a 0x (or 0X) prefix or a leading zero, even if the default radix is hexadecimal.

integer.fraction

A 32-bit floating point number.

'cccc' The ASCII value of up to 4 characters. \ may be used to escape a '.

< name

The value of *name*, which is either a variable name or a register name. *Adb* maintains several variables (see "Variables") named by single letters or digits. If *name* is a register name, the value of the register is obtained from the system header in *corfil*. Register names are those printed by the \$r command.

symbol A symbol is a sequence of upper or lower-case letters, underscores and digits, not starting with a digit. The backslash character ("\") may be used to escape other characters. The value of the symbol is taken from the symbol table in objfil. If symbol is not defined, adb searches for _symbol, the internal name of the entry point of the C function symbol. If _symbol is not defined, adb searches for _symbol, the internal name of the C variable symbol or the data area of the C function symbol.

routine.name

The address of the variableiable name in the specified C routine. Both routine and name are symbols. If name is omitted, the value is the address of the most recently started C stack frame corresponding to routine (see "Bugs").

(exp) The value of the expression exp.

Monadic operators

- *exp > The contents of the location addressed by exp in corfil.
- @exp The contents of the location addressed by exp in objfil.
- exp Integer negation.
- ~exp Bitwise complement.
- #exp Logical negation.

Dyadic operators are left-associative and are less binding than monadic operators.

- el + e2 Integer addition.
- eI e2 Integer subtraction.
- e/*e2 Integer multiplication.
- e1%e2 Integer division.
- e/&e2 Bitwise conjunction.
- el | e2 Bitwise disjunction.
- e1#e2 E1 rounded up to the next multiple of e2.

COMMANDS

Most commands consist of a verb followed by a modifier or list of modifiers. (The commands "?" and "/" may be followed by "*"; see "Addresses" for further details.) The following verbs are available:

- ?f Locations starting at address in objfil are printed according to the format f. Dot is incremented by the sum of the increments for each format letter (q.v.).
- If Locations starting at address in corfil are printed according to the format f and dot is incremented as for "?".
- = f The value of address itself is printed in the styles shown by the format f. (For i, format "?" is printed for the parts of the instruction that refer to subsequent words.)

A format consists of one or more characters specifying a style of printing. Each format character may be preceded by a decimal integer that is a repeat count for the format character. While stepping through a format, dot is incremented by the amount given for each format letter. If no format is given, the last format is used. The format letters available are as follows:

- o 2 Print 2 bytes in octal. All octal numbers output by adb are preceded by 0.
- O 4 Print 4 bytes in octal.
- q 2 Print in signed octal.
- Q 4 Print long signed octal.
- d 2 Print in decimal.
- D 4 Print long decimal.
- x 2 Print 2 bytes in hexadecimal.
- X 4 Print 4 bytes in hexadecimal.
- u 2 Print as an unsigned decimal number.
- U 4 Print long unsigned decimal.
- f 4 Print the 32-bit value as a floating-point number.
- F 8 Print double floating point.
- b 1 Print the addressed byte in octal.
- c 1 Print the addressed character.
- C 1 Print the addressed character using the standard escape convention where control characters are printed as ^X and the delete character is printed as ^?.
- s n Print the addressed characters until a zero character is reached.
- S n Print a string using the X escape convention (n is the length of the string including its zero terminator). See C above.
- Y 4 Print 4 bytes in date format (see ctime(3)).
- in Print as machine instructions (n is the number of bytes occupied by the instruction). This style of printing causes variables 1 and 2 to be set to the offset parts of the source and destination, respectively.
- a 0 Print the value of *dot* in symbolic form. Symbols are checked to ensure that they have the types shown below.
 - / local or global data symbol
 - ? local or global text symbol
 - = local or global absolute symbol
- Print the addressed value in symbolic form using the same rules for symbol lookup as a.
- t 0 Tabs to the next appropriate tab stop when preceded by an integer. For example, 8t moves to the next eight-space tab stop.
- r 0 Print a space.
- n 0 Print a newline.
- "..." 0 Print the enclosed string.
- ^ Dot is decremented by the current increment. Nothing is printed.
- + Dot is incremented by 1. Nothing is printed.
- Dot is decremented by 1. Nothing is printed.

newline

Repeat the previous command with a count of 1.

[?/]I value mask

Words starting at *dot* are masked with *mask* and compared with *value* until a match is found. If L is used then the match is for 4 bytes at a time instead of 2. If no match is found, *dot* is unchanged; otherwise, *dot* is set to the matched location. If *mask* is omitted, -1 is used.

[?/]w value ...

Write the 2-byte value into the addressed location. If the command is W, write 4 bytes. Odd addresses are not allowed when writing to the subprocess address space.

[?/]m b/ e/ f/[?/]

New values for (b1, e1, f1) are recorded. If fewer than three expressions are given, the remaining map parameters are left unchanged. If the "?" or "/" is followed by "*", the second segment (b2, e2, f2) of the mapping is changed. If the list is terminated by "?" or "/", the file (objfil) or corfil, respectively) is used for later requests. (Thus, for example, "/m?" will cause "/" to refer to objfil.)

> name Dot is assigned to the variable or register named.

! A shell (/bin/sh) is called to read the rest of the line following "!".

\$modifier

Miscellaneous commands. The available modifiers are:

- Read commands from the file f. If this command is executed in a file, further commands in the file are not seen. If f is omitted, the current input stream is terminated. If a count is given, and is zero, the command will be ignored. The value of the count will be placed in variable 9 before the first command in f is executed.
- < < f Similar to < except it can be used in a file of commands without causing the file to be closed. Variable 9 is saved during the execution of this command, and restored when it completes. There is a (small) finite limit to the number of < < files that can be open at once.
- > f Append output to the file f which is created if it does not exist. If f is omitted, output is returned to the terminal.
- ? Print process id, the signal that caused stoppage or termination, as well as the registers as **Sr**. This is the default if *modifier* is omitted.
- Print the general registers and the instruction addressed by pc. Dot is set to pc. n\$r prints register n only. n, c\$r prints registers n + c 1, ..., n.
- f Print the floating point registers in hexadecimal and decimal. Register pairs are also formatted as doubles.
- b Print all breakpoints and their associated counts and commands.
- c Backtrace the call stack, starting either at address (if address is given), or at the most recently activated frame (if address is not given). If count is given, only the first count frames are printed.
- C Like c; additionally print the auto area or the last 64 words of it, if it is large. The last word printed corresponds to the first auto allocated by the C compiler.
- d Set the default radix to address and report the new value. Note that address is interpreted in the (old) current radix. Thus "10\$d" never changes the default radix. To make decimal the default radix, use "0t10\$d".
- e The names and values of external variables are printed.
- w Set the page width for output to address (default 80).
- s Set the limit for symbol matches to address (default 255).
- o All integers input are regarded as octal.
- **q** Exit from adb.
- v Print all non-zero variables in octal.
- m Print the address map.
- p (Kernel debugging) Change the current kernel memory mapping to map the designated user structure to the address given by the symbol <u>u</u>. The address argument is the address of the user's user page table entries.

:modifier

Manage a subprocess. Available modifiers are:

bc Set breakpoint at address. The breakpoint is executed count-1 times before causing a stop. Each time the breakpoint is encountered the command c is executed. If this command is omitted or sets dot to zero, the breakpoint causes a

stop.

- d Delete breakpoint at address.
- Run objfil as a subprocess. If address is given explicitly, the program is entered at this point; otherwise the program is entered at its standard entry point. Count specifies how many breakpoints are to be ignored before stopping. Arguments to the subprocess may be supplied on the same line as the command. An argument starting with < or > causes the standard input or output to be established for the command.
- The subprocess is continued with signal s (see sigvec(2)). If address is given, the subprocess is continued at this address. If no signal is specified, the signal that caused the subprocess to stop is sent. Breakpoint skipping is the same as for r.
- As for c, except that the subprocess is single-stepped *count* times. If there is no current subprocess, *objfil* is run as a subprocess as for r. Here no signal can be sent; the remainder of the line is treated as arguments to the subprocess.
- k The current subprocess, if any, is terminated.

VARIABLES

Adb provides several variables. Named variables are set initially by adb but are not used subsequently. Numbered variables are reserved for communication as follows:

- 0 The last value printed.
- 1 The last offset part of an instruction source.
- The previous value of variable 1.
- 9 The count on the last \$< or \$< < command.

On entry, the following are set from the system header in the *corfil*. If *corfil* does not appear to be a core file, these values are set from *objfil*.

- b The base address of the data segment.
- d The data segment size.
- e The entry point.
- m The "magic number" (0407, 0410 or 0413) (see a.out(5)).
- s The stack segment size.
- t The text segment size.

ADDRESSES

The address in a file associated with a written address is determined by a mapping associated with that file. Two triples, (b1, e1, f1) and (b2, e2, f2), represent each mapping, and the *file address* corresponding to a written address is calculated as follows.

```
b1 \le address \le e1 = b, file address = address + f1 - b1, otherwise, b2 \le address \le e2 = b, file address = address + f2 - b2,
```

otherwise, the requested address is not legal. Sometimes (e.g. for programs with separated I and D space) the two segments for a file may overlap. If a ? or / is followed by an * then only the second triple is used.

The initial setting of both mappings is suitable for normal a.out and core files. If either file is not of the kind expected then, for that file, bl is set to 0, el is set to the maximum file size and fl is set to 0; in this way the whole file can be examined with no address translation.

FILES

a.out

SEE ALSO

cc(1), dbx(1), ptrace(2), a.out(5), core(5)

"Using ADB to Debug the Kernel", by S. Leffler and W. Joy, in the UNIX System Manager's Manual

"IBM/4.3 Linkage Convention" in Volume II, Supplementary Documents

DIAGNOSTICS

"Adb" is printed when there is no current command or format. Adb complains about inaccessible files, syntax errors, abnormal termination of commands, etc. Exit status is 0, unless the last command failed or returned nonzero status.

BUGS

Since no shell is invoked to interpret the arguments of the :r command, the customary wild-card and variable expansions cannot occur.

The **Sc** and **SC** commands print incorrect values for arguments of class register. Backtracing may fail entirely or print erroneous values if execution of the most recently called function has not proceeded past prolog code (e.g. when a breakpoint stops execution on an entry point). Single step a few times, and reissue the **Sc** or **SC**.

Postmortem stack backtraces on a corfil require an objfil also.

The \$C command lists the values of automatic variables, but does not display their names; the symbol table does not currently contain the names of automatic variables. The absence of automatic-variable names in the symbol table also means that expressions of the form "routine.name" are not supported.

The adb scripts in /usr/lib/adb for kernel debugging probably do not work.

Setting of variables 1 and 2 is not implemented for i format.

NOTE

Because of the asynchronous nature of loads and stores on a machine with an Advanced Processor Card, the CPU may have executed one or more instructions beyond the one that causes a program check. This means that the iar will be pointing to the instruction that would have been executed had the program check not occurred, and not the instruction that caused the program check. The \$r (or \$) command will display the actual exception packets, if any, in pseudo-instruction form. This may aid in locating the actual source of the error. See the *IBM RT PC Hardware Technical Reference, Volume I*, SV21-8024, for more information on exception packets.

NAME

aedjournal - display commands in a log file

SYNOPSIS

aedjournal file

DESCRIPTION

Although there is no debugging facility as such supplied with the C subroutine interface to the display, you can use VI_Login and VI_Logout (see log(3G)) with aedjournal to help follow your application program's actions. Aedjournal deciphers a file produced by VI_Login and reports to standard output all orders passed to the display. Standard output may be redirected as usual. You may inspect this output to discover unintended results.

Beware of the length of logged files. It is easy to generate thousands of display orders for a seemingly simple picture; thus, try to log the smallest group of orders likely to contain the bug. The log routines may be called several times in one application to produce several files of orders, requiring only that each call to *VI Login* provide a distinct file name.

NOTE

Aedjournal applies only to the IBM Academic Information Systems experimental display.

SEE ALSO

aedrunner(1), init(3G), log(3G), ibmaed(4)

AEDRUNNER(1) AEDRUNNER(1)

NAME

aedrunner - execute graphics commands in a log file

SYNOPSIS

aedrunner file ...

DESCRIPTION

Aedrunner executes the logged orders contained in the specified file, which must have been created with VI_Login and VI_Logout (see log(3G)). Aedrunner terminates upon discovery of any error or inconsistency in the file. All additional files which were needed when the log file was constructed must be available in the current directory. Such files are any font, image, or cursor definition files you may have used. Images, cursors, or tiles defined from memory are handled by the log routines and do not require regeneration.

NOTE

Aedrunner applies only to the IBM Academic Information Systems experimental display.

FILES

```
/usr/lib/aed/whim.aed
/usr/lib/aed/pcfont.fnt
/dev/aed
/dev/console
```

SEE ALSO

aedjournal(1), init(3G), log(3G), ibmaed(4)

BUGS

If the log file is corrupt in any way, aedrunner merely terminates. Use aedjournal to examine the file.

ANDREW(1) ANDREW(1)

NAME

andrew - start Xibm, cwm, console, and typescript

SYNOPSIS

```
andrew [server] [display ...]
```

DESCRIPTION

This is a shell script to be used in conjunction with "The IBM Andrew Toolkit User's Guide" by anyone who wants to learn how the Andrew File System works.

To use it, set the following environment variables:

DISPLAY :0 CLASSPATH /usr/andrew/dlib/be2 BE2WM x11

and copy two files, preferences and .Xdefaults, from /usr/guest/guest/andrew to your home directory.

Specifically, andrew starts up the Xibm server, cwm window manager, and two andrew applications, console and typescript, on the specified server and display.

Server is the server to uniquely identify a particular server. It consists of a colon followed by one digit in the range of 0-7; for example, :0. If no server is specified, 0 is used.

The following displays are recognized:

8514 The IBM 8514 PS/2 Color Graphics Display Adapter 8514/A

vga The IBM Video Graphics Array (VGA) Display

mpel The IBM 5081 Display with MegaPel adapter

ega The IBM 5154 Enhanced Color Display with adapter

apa16 The IBM 6155 Extended Monochrome Graphics Display

aed The IBM Academic Information Systems Experimental Display

If no display is specified, andrew starts on the first available display chosen from the list above, searching from top to bottom.

EXAMPLES

andrew vga 8514 andrew mpel

VARIABLES

If no server number is supplied on the command line, andrew looks for the environment variable DISPLAY and, if it is defined, uses it for server. If a server number is given on the command line, it overrides the value defined in the DISPLAY variable in the shell environment.

FILES

```
/usr/guest/guest/andrew/andrew
/usr/guest/guest/andrew/preferences
/usr/guest/guest/andrew/.Xdefaults
```

SEE ALSO

```
cmenu(1), cwm(1), Xibm(1)
"The Andrew Toolkit User's Guide"
```

AS(1)

NAME

as - assembler

SYNOPSIS

as
$$[-L][-V][-W][-R][-D][-T][-t$$
 directory $[-o][-o][-w][-w]$

DESCRIPTION

As assembles the named files, or the standard input if no file name is specified. The available flags are:

- -L Save defined labels beginning with an "L", which are normally discarded to save space in the resultant symbol table. The compilers generate such temporary labels.
- -V Use virtual memory rather than a temporary file for the interpass temporary file.
- -W Turn off all warning error reporting.
- R Make initialized data segments read-only by concatenating them to the text segments. This obviates the need to run editor scripts on assembly code to make initialized data read-only and shared.
- -D Print assembler debugging information and dump the symbol table, provided the assembler has been compiled with DEBUG defined.
- -T Print the token file, provided the assembler has been compiled with DEBUG defined.
- -t Specify a directory to receive the temporary file other than the default /tmp, provided the -V flag is not set.
- $-\mathbf{o}$ Place the output of the assembly in the file *objfile*; if that is omitted, *a.out* is used.

All undefined symbols in the assembly are treated as global. Flags $-\mathbf{J}$ and $-\mathbf{d}$ are ignored.

FILES

/tmp/as* default temporary files a.out default resultant object file

SEE ALSO

adb(1), dbx(1), ld(1), nm(1), a.out(5)

"Assembler Reference Manual for IBM/4.3" in Volume II, Supplementary Documents

BITPRT(1)

NAME

bitprt - capture the image on a bitmap display and print it on an IBM printer

SYNOPSIS

```
bitprt display [-t][-Pxx][-Lparms]
```

DESCRIPTION

Bitprt captures a "snapshot" of the image on a display device (specified as indicated below), and routes it to a printer (IBM 5152 Graphics Printer, IBM 4201 Proprinter, or IBM 3812 Page-printer) or to standard output.

A display is one of the following:

- -a selects the IBM Academic Information Systems experimental display;
- -8 selects the apa8 (IBM 6153 Advanced Monochrome Graphics Display);
- -8c selects the apa8c (IBM 6154 Advanced Color Graphics Display);
- -16 selects the apa16 (IBM 6155 Extended Monochrome Graphics Display).

Options are:

- t send output to standard output.
- -Pxx send output to printer xx which corresponds to an entry in /etc/printcap. The default is taken from the PRINTER environment variable, if it exists; otherwise |p| is used.
- -Lparms

```
pass parameter parms to lpr(1). This may be a single parameter (e.g., -L-h) or a string (e.g., -L"-h -r -m").
```

NOTES

The following printers are supported:

IBM 3812 Pageprinter

IBM 4201 Proprinter

IBM 5152 Graphics Printer

Bitprt is not supported on the IBM 6152 Academic System.

FILES

```
/etc/printcap
/dev/bus
/dev/lp
```

SEE ALSO

```
dumpaed(1), dumpapa16(1), dumpapa8(1), dumpapa8c(1), lpr(1), scale(1), ibm6153(4), ibm6154(4), ibm6155(4), ibmaed(4), lp(4), lpfilter(8R)
```

BUGS

Only the first color plane for the IBM 6154 Advanced Color Graphics display is captured.

CC(1)

NAME

cc - default C compiler

SYNOPSIS

cc [option] ... file ...

DESCRIPTION

As shipped, cc is a symbolic link to |bin/hc|. Your system administrator can change this link to |bin/pcc|, if you require pcc to be the default C compiler.

Note: If you wish to leave hc as the default compiler but object to the increased number of warning messages, you can install the following script in |bin|/cc:

#! /bin/sh exec /bin/hc -w "\$@"

FILES

/bin/cc /bin/hc /bin/pcc

SEE ALSO

hc(1), pcc(1)

COLPRO(1) COLPRO(1)

NAME

colpro - column filter for IBM 4201 Proprinter

SYNOPSIS

colpro [- bfh]

DESCRIPTION

Colpro reads the standard input and writes the standard output. It performs the line overlays implied by reverse line feeds (<Esc>-7 in ASCII) and by forward and reverse half line feeds (<Esc>-9 and <Esc>-8). Colpro is particularly useful for filtering multiple-column output made with the '.rt' command of nroff and output resulting from use of the tbl(1) preprocessor. Colpro is intended for use with prfl(1).

Although colpro accepts half line motions in its input, it normally does not emit them on output. Instead, text appearing between lines is moved to the next lower full line boundary. This treatment can be suppressed by the $-\mathbf{f}$ (fine) option. In this case the output from colpro may contain forward half line feeds < Esc > -9, but will still never contain either kind of reverse line motion.

If the $-\mathbf{b}$ option is given, *colpro* assumes the output device in use is not capable of backspacing. In this case, if several characters are to appear in the same place, only the last one read will be taken.

Colpro may also be used with the IBM 5152 Graphics Printer. Note that even though the graphics printer cannot back up, the -b option should not be used since backspaces tell the post filter to use underlining.

< Esc > sequences are treated according to one of three rules:

- 1) < Esc > -7,8,9 are line positioning commands and are treated as above.
- 2) < Esc > -Ax sequences are assumed to be printer control sequences (initialize printer, start bold, stop bold, etc.). These are echoed to standard output and are treated as if they have "0" width.
- 3) All other < Esc> sequences are assumed to be 3 characters long (i.e. < Esc> c1 c2), and are echoed to standard output as is (even if one of the characters is a control character). The filter assumes the three-character escape sequence specifies a single printable character.

If the -h option is given, *colpro* converts white space to tabs to shorten printing time.

No control characters are passed to the output except space, backspace, tab, return, newline, < Esc > (033) followed by any 2 characters (except 7,8 or 9) and VT (013). This last character is an alternate form of full reverse line feed, for compatibility with some other hardware conventions. All other non-printing characters are ignored.

Colpro is provided primarily for use by proff.

SEE ALSO

col(1), nroff(1), prfl(1), proff(1), tbl(1), troff(1)

BUGS

Colpro cannot back up more than 128 lines.

Line length is limited to 800 characters, including backspaces.

Colpro cannot handle more than 127 unique escape sequences in any 128 contiguous lines of printed text (excluding < Esc > -7,8,9).

DATE(1)

NAME

date - print and set the date

SYNOPSIS

```
date [ -n ] [ -u ] [ yymmddhhmm [ . ss ] ]
```

DESCRIPTION

If no arguments are given, the current date and time are printed. Providing an argument will set the desired date. Only the superuser can set the date. The -u flag is used to display or set the date in GMT (universal) time. Yy represents the last two digits of the year; the first mm is the month number; dd is the day number; hh is the hour number (24 hour system); the second mm is the minute number; ss is optional and represents the seconds. For example:

date 8506131627

sets the date to June 13 1985, 4:27 PM. The year, month and day may be omitted; the default values will be the current ones. The system operates in GMT. Date takes care of the conversion to and from local standard and daylight-saving time.

If timed(8) is running to synchronize the clocks of machines in a local area network, date sets the time globally on all those machines unless the -n option is given.

FILES

/usr/adm/wtmp to record time-setting. In /usr/adm/messages, date records the name of the user setting the time.

SEE ALSO

gettimeofday(2), utmp(5), timed(8),

TSP: The Time Synchronization Protocol for UNIX 4.3BSD, R. Gusella and S. Zatti

DIAGNOSTICS

Exit status is 0 on success, 1 on complete failure to set the date, and 2 on successfully setting the local date but failing globally.

'You are not superuser: date not set' if you try to change the date but are not the super-user. Occasionally, when *timed* synchronizes the time on many hosts, the setting of a new time value may require more than a few seconds. On these occasions, *date* prints: 'Network time being set'. The message 'Communication error with timed' occurs when the communication between *date* and *timed* fails.

NAME

dbx - debugger

SYNOPSIS

$$dbx [-r][-i][-I dir][-c file][objfile[coredump]]$$

DESCRIPTION

Dbx is a tool for source level debugging and execution of programs. The *objfile* is an object file produced by a compiler with the appropriate flag (usually -g) specified to produce symbol information in the object file. Currently, f77(1), hc(1), pcc(1), and pp(1) produce the appropriate source information. The assembler-machine level facilities of dbx can be used on any program.

The object file contains a symbol table that includes the name of the all the source files translated by the compiler to create it. These files can be perused while using dbx.

If a file named "core" exists in the current directory or a *coredump* file is specified, dbx can be used to examine the state of the program when it faulted.

If the file ".dbxinit" exists in the current directory then the debugger commands in it are executed. Dbx also checks for a ".dbxinit" in the user's home directory if there isn't one in the current directory.

The command line options and their meanings are:

- -r Execute objfile immediately. If execution terminates successfully dbx exits. Otherwise dbx reports the reason and the user is offered the option of continuing with dbx or quitting. Dbx will read from "/dev/tty" when -r is specified and standard input is not a terminal.
- -i Force dbx to act as though standard input is a terminal.
- -I dir Add dir to the list of directories that are searched when looking for a source file. Normally dbx looks for source files in the current directory and in the directory where objfile is located. The directory search path can also be set with the use command.
- -c file Execute the dbx commands in the file before reading from standard input.

Unless $-\mathbf{r}$ is specified, dbx just prompts and waits for a command.

Execution and Tracing Commands

```
run [args] [ < filename] [ > filename] rerun [args] [ < filename] [ > filename]
```

Start executing objfile, passing args as command line arguments; < or > can be used to redirect input or output in a shell-like manner. When **rerun** is used without any arguments the previous argument list is passed to the program; otherwise it is identical to **run**. If objfile has been written since the last time the symbolic information was read in, dbx will read in the new information.

trace [in procedure/function] [if condition]
trace source-line-number [if condition]
trace procedure/function [in procedure/function] [if condition]
trace expression at source-line-number [if condition]
trace variable [in procedure/function] [if condition]

Have tracing information printed when the program is executed. A number is associated with the command so that tracing can be turned off with the delete command.

DBX(1)

The first argument describes what is to be traced. If it is a *source-line-number*, then the line is printed immediately prior to being executed. Source line numbers in a file other than the current one must be preceded by the name of the file in quotes and a colon, e.g. "mumble.p":17.

If the argument is a procedure or function name then every time it is called, information is printed telling what routine called it, from what source line it was called, and what parameters were passed to it. In addition, its return is noted, and if it's a function then the value it is returning is also printed.

If the argument is an expression with an at clause then the value of the expression is printed whenever the identified source line is reached.

If the argument is a variable then the name and value of the variable is printed whenever it changes. Execution is substantially slower during this form of tracing.

If no argument is specified then all source lines are printed before they are executed. Execution is substantially slower during this form of tracing.

The clause "in procedure/function" restricts tracing information to be printed only while executing inside the given procedure or function.

Condition is a boolean expression and is evaluated prior to printing the tracing information; if it is false then the information is not printed.

stop if condition

stop at source-line-number [if condition]

stop in procedure/function [if condition]

stop variable [if condition]

Stop execution when the given line is reached, procedure or function called, variable changed, or condition true.

status [> filename]

Print out the currently active trace and stop commands.

delete command-number ...

The traces or stops corresponding to the given numbers are removed. The numbers associated with traces and stops are printed by the status command.

catch signal-number catch signal-name ignore signal-number ignore signal-name

Start or stop trapping a signal before it is sent to the program. This is useful when a program being debugged handles signals such as interrupts. A signal may be specified by number or by a name (e.g., SIGINT). Signal names are case insensitive and the "SIG" prefix is optional. By default all signals are trapped except SIGCONT, SIGCHILD, SIGALRM and SIGKILL.

cont

cont signal-number

cont signal-name

Continue execution from where it stopped. If a signal is specified, the process continues as though it received the signal. Otherwise, the process is continued as though it had not been stopped.

Execution cannot be continued if the process has "finished", that is, called the standard procedure "exit". Dbx does not allow the process to exit, thereby letting the user examine the program state.

step Execute one source line.

next Execute up to the next source line. The difference between this and step is that if the line contains a call to a procedure or function the step command will stop at the beginning of that block, while the next command will not.

return [procedure]

Continue until a return to procedure is executed, or until the current procedure returns if none is specified.

call procedure(parameters)

Execute the object code associated with the named procedure or function.

Printing Variables and Expressions

Names are resolved first using the static scope of the current function, then using the dynamic scope if the name is not defined in the static scope. If static and dynamic searches do not yield a result, an arbitrary symbol is chosen and the message "[using qualified name]" is printed. The name resolution procedure may be overridden by qualifying an identifier with a block name, e.g., "module.variable". For C, source files are treated as modules named by the file name without ".c".

Expressions are specified with an approximately common subset of C and Pascal syntax. Indirection can be denoted using either a prefix "*" or a postfix "^" and array expressions are subscripted by brackets ("[]"). The field reference operator (".") can be used with pointers as well as structures, unions, and records, making the C operator "->" unnecessary (although it is supported).

Types of expressions are checked; the type of an expression may be overridden by using "type-name(expression)". When there is no corresponding named type the special constructs "&type-name" and "\$\$tag-name" can be used to represent a pointer to a named type or C structure tag.

assign variable = expression

Assign the value of the expression to the variable.

dump [procedure] [> filename]

Print the names and values of variables in the given procedure, or the current one if none is specified. If the procedure given is ".", then the all active variables are dumped.

print expression [, expression ...]

Print out the values of the expressions.

whatis name

Print the declaration of the given name, which may be qualified with block names as above.

which identifier

Print the full qualification of the given identifier, i.e. the outer blocks that the identifier is associated with.

up [count]

down [count]

Move the current function, which is used for resolving names, up or down the stack count levels. The default count is 1.

where Print out a list of the active procedures and functions.

whereis identifier

Print the full qualification of all the symbols whose name matches the given identifier. The order in which the symbols are printed is not meaningful.

Accessing Source Files

```
/regular expression[/]
?regular expression[?]
```

Search forward or backward in the current source file for the given pattern.

edit [filename]

edit procedure/function-name

Invoke an editor on *filename* or the current source file if none is specified. If a *procedure* or *function* name is specified, the editor is invoked on the file that contains it. Which editor is invoked by default depends on the installation. The default can be overridden by setting the environment variable EDITOR to the name of the desired editor.

file [filename]

Change the current source file name to *filename*. If none is specified then the current source file name is printed.

func [procedure/function]

Change the current function. If none is specified then print the current function. Changing the current function implicitly changes the current source file to the one that contains the function; it also changes the current scope used for name resolution.

```
list [source-line-number [, source-line-number]] list procedure|function
```

List the lines in the current source file from the first line number to the second inclusive. If no lines are specified, the next 10 lines are listed. If the name of a procedure or function is given lines n-k to n+k are listed where n is the first statement in the procedure or function and k is defined by \$listwindow.

use directory-list

Set the list of directories to be searched when looking for source files.

Command Aliases and Variables

```
alias
alias name name
alias name "string"
alias name (parameters) "string"
```

When commands are processed, dbx first checks to see if the word is an alias for either a command or a string. If it is an alias, then dbx treats the input as though the corresponding string (with values substituted for any parameters) had been entered. For example, to define an alias "rr" for the command "rerun", one can say:

alias rr rerun

To define an alias called "b" that sets a stop at a particular line one can say:

alias b(x) "stop at x"

Subsequently, the command "b(12)" will expand to "stop at 12". Alias with no arguments prints the alias definition list.

set name [= expression]

The set command defines values for debugger variables. The names of these variables cannot conflict with names in the program being debugged, and are expanded to the corresponding expression within other commands. The following variables have a special meaning:

\$frame

Setting this variable to an address causes dbx to use the stack frame pointed to by the address for doing stack traces and accessing local variables.

\$hexchars

\$hexints

\$hexoffsets

\$hexstrings

When set, dbx prints characters, integers, offsets from registers, or character pointers respectively in hexadecimal.

\$listwindow

The value of this variable specifies the number of lines to list around a function or when the list command is given without any parameters. Its default value is 10.

\$mapaddrs

Setting (unsetting) this variable causes dbx to start (stop) mapping addresses. As with "\$frame", this is useful for kernel debugging.

\$unsafecall

\$unsafeassign

When "sunsafecall" is set, strict type checking is turned off for arguments to subroutine or function calls (e.g. in the call statement). When "sunsafeassign" is set, strict type checking between the two sides of an assign statement is turned off. These variables should be used only with great care, because they severely limit dbx's usefulness for detecting errors.

unalias name

Remove the alias with the given name.

unset name

Delete the debugger variable associated with name.

Machine Level Commands

```
tracei [address] [if cond]
tracei [variable] [at address] [if cond]
stopi [address] [if cond]
stopi [at] [address] [if cond]
```

Turn on tracing or set a stop using a machine instruction address.

stepi

nexti Single step as in step or next, but do a single instruction rather than source line.

address, address | [mode] address | [count] [mode]

Print the contents of memory starting at the first address and continuing up to the second address or until count items are printed. If the address is ".", the address following the one printed most recently is used. The mode specifies how memory is to be printed; if it is omitted the previous mode specified is used. The initial mode is "X". The following modes are supported:

- i print the machine instruction
- d print a short word in decimal
- D print a long word in decimal
- o print a short word in octal
- O print a long word in octal
- x print a short word in hexadecimal
- X print a long word in hexadecimal
- b print a byte in octal
- c print a byte as a character
- s print a string of characters terminated by a null byte
- f print a single precision real number
- g print a double precision real number

Symbolic addresses are specified by preceding the name with an "&". Registers are denoted by "\$rN" where N is the number of the register. Addresses may be expressions made up of other addresses and the operators "+", "-", and indirection (unary "*").

Miscellaneous Commands

help Print out a synopsis of dbx commands.

quit Exit dbx.

sh command-line

Pass the command line to the shell for execution. The SHELL environment variable determines which shell is used.

source filename

Read dbx commands from the given filename.

FILES

a.out

object file

.dbxinit

initial commands

SEE ALSO

f77(1), hc(1), pcc(1), pp(1)

COMMENTS

Dbx suffers from the same "multiple include" malady as did sdb. If you have a program consisting of a number of object files and each is built from source files that include header files, the symbolic information for the header files is replicated in each object file. Since about one debugger start-up is done for each link, having the linker (ld) re-organize the symbol information would not save much time, though it would reduce some of the disk space used.

This problem is an artifact of the unrestricted semantics of #include's in C; for example, an include file can contain static declarations that are separate entities for each file in which they are included.

Some problems remain with the support for individual languages. Fortran problems include: inability to assign to logical, logical*2, complex and double complex variables; inability to represent parameter constants which are not type integer or real; peculiar representation for the values of dummy procedures (the value shown for a dummy procedure is actually the first few bytes of the procedure text; to find the location of the procedure, use "&" to take the address of the variable).

The kernel debugging option -k is not implemented.

DOSREAD(1) DOSREAD(1)

NAME

dosread - read, write, dir, delete on PC-DOS diskette

SYNOPSIS

```
dosread [-xdtw][-v][-R][-a][-I][-ffloppy][-ppart][-Tinfo][pattern]...
doswrite [-xdtw][-v][-a][-isize][-b][-ffloppy][-Ppart]files...
dosdir [-xdtw][-v][-R][-a][-ffloppy][-Ppart][pattern]...
dosdel [-xdtw][-v][-a][-ffloppy][-Ppart][pattern]...
```

DESCRIPTION

Dosread allows manipulation of diskettes or fixed disks in IBM PC-DOS format. Several switches control its actions:

- -x extract files (default, if invoked with a name of dosread). All the files that match pattern are extracted from the diskette. The pattern can contain pattern-match characters (*, ?, [...]) which should be enclosed in quotation marks to prevent the shell from expanding them prematurely. If no pattern is specified, all files are used.
- -d deletes files (default, if invoked with a name of *dosdel*). All files that match *pattern* are deleted from the diskette. *Pattern* is treated as for *dosread*.
- -t table of contents (default, if invoked with a name of dosdir). The table of contents of the diskette is printed. Pattern is treated as for dosread.
- -w writes files (default, if invoked with a name of doswrite). All the files specified by files are written onto the diskette. Note that any pattern characters should NOT be enclosed in quotation marks (e.g. *.c is acceptable, but "*.c" is not).
- requests a more verbose printout. For dosread, doswrite and dosdel, this involves printing a short line giving the action taken. In dosdir, it causes more verbose output (similar to the dir command in PC-DOS) to be printed.
- -a causes files to be treated as ASCII. With dosread, this removes the CR in CR/LF sequences and treats ^Z as an end-of-file indication. With doswrite, it inserts a CR before LF and appends a ^Z to the end of the file.
- -f specifies that the following argument contain the name of the file containing the PC-DOS file system, instead of the default diskette drive.
- isize specifies that the diskette be initialized. Size is the size of the diskette (either 360 or 1200) in units of 1024 bytes. If no size is specified, then the size is determined heuristically.
- -I ignores information in the boot sector and determines diskette type solely from FAT type code.
- -b specifies that an RT PC boot block is to be written into block zero of the diskette. The ROS IPL procedure attempts to load the first file on the diskette. This file must occupy contiguous blocks. To insure this, specify the -i option and the file in the same doswrite command as the -b option. The file must be in self-relocating a out format. If the -b option is not specified when a diskette is initialized, a PC/DOS boot block will be installed (if it exists) from the file /usr/mdec/fd*pcdosboot where * is the size of the diskette in K byte units.

-Ppart

specifies that *floppy* is to be treated as a PC-DOS fixed disk with a PC-DOS partition table. If *part* is specified, then that partition is used; otherwise, the active partition is used.

- -R recursively processes any PC-DOS subdirectories encountered.
- Tinfo

h

allows specification of new diskette parameters (info values are specified as a comma-

DOSREAD(1)

DOSREAD(1)

separated list), and are: fat_type, fat_size, sectors, heads, dir_sectors, cluster_size, tracks, size, and fat_copes. Each may be specified as either a decimal or hex (with leading 0x) constant. The *info* items and order may change in a later version of *dosread*.

BUGS

Doswrite and dosdel still do not process subdirectories.

Dosread does not handle 16-bit FAT entries.

FILES

/dev/fd0 for the default case -- the diskette drive (360k or 1.2Mb) /usr/mdec/fd*dosboot -- for the various boot blocks

SEE ALSO

Disk Operating System Version 3.00 Technical Reference, 6322677

DUMPAED(1)

NAME

dumpaed - dump aed display memory as a binary file

SYNOPSIS

dumpaed

DESCRIPTION

Dumpaed copies the contents of the aed display memory to the standard output for redirection to a binary file or for processing as a bitmap image.

The output consists of a two-integer header (horizontal image size in pixels; vertical image size in pixels) followed by the image itself in left-to-right, top-to-bottom order, (horizontal_pixels / 8) bytes per scanline. For the *aed*, the horizontal size is 1024; the vertical size is 800.

Dumpaed uses the whim.aed microcode and subroutine calls from the aed subroutine library to do its work. If whim.aed is not loaded when dumpaed is invoked, |dev|aed will be opened and whim.aed loaded, resulting in a glass tty microcode reload when dumpaed completes. (See ibmaed(4)).

Dumpaed is provided primarily for use by bitprt(1).

NOTES

Dumpaed applies only to the IBM Academic Information Systems experimental display.

FILES

```
/dev/aed
/dev/bus
/usr/lib/aed/whim.aed
```

SEE ALSO

```
bitprt(1), lpr(1), scale(1), ibmaed(4), lp(4), lpfilter(8R)
"The C Subroutine Interface for the IBM Academic Information Systems Experimental Display"
in Volume II, Supplementary Documents
```

BUGS

Reloading of the glass tty microcode causes the aed screen to be cleared.

DUMPAPA16(1)

NAME

dumpapa16 - dump apa16 display memory as a binary file

SYNOPSIS

dumpapa 16

DESCRIPTION

Dumpapal6 copies the contents of the apal6 display memory to the standard output for redirection to a binary file or for processing as a bitmap image.

The output consists of a two-integer header (horizontal image size in pixels; vertical image size in pixels) followed by the image itself in left-to-right, top-to-bottom order, (horizontal_pixels / 8) bytes per scanline. For the apa16 the horizontal size is 1024; the vertical size is 768.

Before any image dumping is begun, /dev/apa16 is opened to be sure the device is present.

Dumpapa16 is provided primarily for use by bitprt(1).

FILES

/dev/apa16 /dev/bus

SEE ALSO

bitprt(1), lpr(1), scale(1), ibm6155(4), lp(4), lpfilter(8R)

BUGS

If the apa16 was in glass-tty mode, the screen is cleared when dumpapa16 completes.

DUMPAPA8(1) DUMPAPA8(1)

NAME

dumpapa8 - dump apa8 display memory as a binary file

SYNOPSIS

dumpapa8

DESCRIPTION

Dumpapa8 copies the contents of the apa8 display memory to the standard output for redirection to a binary file or for processing as a bitmap image.

The output consists of a two-integer header (horizontal image size in pixels; vertical image size in pixels) followed by the image itself in left-to-right, top-to-bottom order, (horizontal_pixels / 8) bytes per scanline. For the apa8 the horizontal size is 720; the vertical size is 512.

Before any image dumping is begun, /dev/apa8 is opened to be sure the device is present.

Dumpapa8 is provided primarily for use by bitprt(1).

FILES

```
/dev/apa8
/dev/bus
```

SEE ALSO

```
bitprt(1), lpr(1), scale(1), ibm6153(4), lp(4), lpfilter(8R)
```

BUGS

If the apa8 was in glass-tty mode, the screen is cleared when dumpapa8 completes.

DUMPAPA8C(1) DUMPAPA8C(1)

NAME

dumpapa8c - dump apa8c display memory as a binary file

SYNOPSIS

dumpapa8c

DESCRIPTION

Dumpapa8c copies the contents of the apa8c display memory to the standard output for redirection to a binary file or for processing as a bitmap image.

The output consists of a two-integer header (horizontal image size in pixels; vertical image size in pixels) followed by the image itself in left-to-right, top-to-bottom order, (horizontal_pixels / 8) bytes per scanline. For the apa8c the horizontal size is 720; the vertical size is 512.

Before any image dumping is begun, |dev|apa8c is opened to be sure the device is present.

Dumpapa8c is provided primarily for use by bitprt(1).

FILES

/dev/apa8c /dev/bus

SEE ALSO

bitprt(1), lpr(1), scale(1), ibm6154(4), lp(4), lpfilter(8R)

BUGS

Hardware limitations allow dumping only one display plane at a time, as controlled by the display adapter's Color Plane Select Register. *Dumpapa8c* assumes it was set correctly.

If the apa8c was in glass-tty mode, the screen is cleared when dumpapa8c completes.

ERROR(1) ERROR(1)

NAME

error - analyze and disperse compiler error messages

SYNOPSIS

error
$$[-n][-s][-q][-v][-t$$
 suffixlist $[-I]$ ignorefile $[-I]$ name $[-I]$

DESCRIPTION

Error analyzes and optionally disperses the diagnostic error messages produced by a number of compilers and language processors to the source file and line where the errors occurred. It can replace the painful, traditional methods of scribbling abbreviations of errors on paper, and permits error messages and source code to be viewed simultaneously without machinations of multiple windows in a screen editor.

Error looks at the error messages, either from the specified file name or from the standard input, and attempts to determine which language processor produced each error message, determines the source file and line number to which the error message refers, determines if the error message is to be ignored or not, and inserts the (possibly slightly modified) error message into the source file as a comment on the line preceding the line to which the error message refers. Error messages which can't be categorized by language processor or content are not inserted into any file, but are sent to the standard output. Error touches source files only after all input has been read. By specifying the -q query option, the user is asked to confirm any potentially dangerous (such as touching a file) or verbose action. Otherwise error proceeds on its merry business. If the -t touch option and associated suffix list is given, error will restrict itself to touch only those files with suffices in the suffix list. Error also can be asked (by specifying -v) to invoke vi(1) on the files in which error messages were inserted; this obviates the need to remember the names of the files with errors.

Error is intended to be run with its standard input connected via a pipe to the error message source. Some language processors put error messages on their standard error file; others put their messages on the standard output. Hence, both error sources should be piped together into error. For example, when using the csh syntax,

make
$$-s$$
 lint $|&$ error $-q$ $-v$

will analyze all the error messages produced by whatever programs make runs when making lint.

Error knows about the error messages produced by: as, cc, ccom, cpp, f77, hc, hccom, ld, lint, make, pc, pcc, pi, pp, ppcom, and DEC Western Research Modula-2. Error knows a standard format for error messages produced by the language processors, so is sensitive to changes in these formats. Some error messages refer to more than one line in more than one file; error will duplicate the error message and insert it at all of the places referenced.

Error will do one of six things with error messages.

synchronize

Some language processors produce short errors describing which file it is processing. *Error* uses these to determine the file name for languages that don't include the file name in each error message. These synchronization messages are consumed entirely by *error*.

discard Error messages from lint that refer to one of the two lint libraries, /usr/lib/llib-lc and /usr/lib/llib-port are discarded, to prevent accidently touching these libraries. Again, these error messages are consumed entirely by error.

mullify Error messages from lint can be nullified if they refer to a specific function, which is known to generate diagnostics which are not interesting. Nullified error messages are not inserted into the source file, but are written to the standard output. The names of functions to ignore are taken from either the file named .errorrc in the users's home directory, or from the file named by the -I option. If the file does not exist, no error messages are nullified. If the file does exist, there must be one function name per line.

ERROR(1) ERROR(1)

not file specific

Error messages that can't be intuited are grouped together, and written to the standard output before any files are touched. They will not be inserted into any source file.

file specific

Error message that refer to a specific file, but to no specific line, are written to the standard output when that file is touched.

true errors Error messages that can be intuited are candidates for insertion into the file to which they refer.

Only true error messages are candidates for inserting into the file they refer to. Other error messages are consumed entirely by error or are written to the standard output. Error inserts the error messages into the source file on the line preceding the line the language processor found in error. Each error message is turned into a one line comment for the language, and is internally flagged with the string "###" at the beginning of the error, and "%%%" at the end of the error. This makes pattern searching for errors easier with an editor, and allows the messages to be easily removed. In addition, each error message contains the source line number for the line the message refers to. A reasonably formatted source program can be recompiled with the error messages still in it, without having the error messages themselves cause future errors. For poorly formatted source programs in free format languages, such as C or Pascal, it is possible to insert a comment into another comment, which can wreak havoc with a future compilation. To avoid this, programs with comments and source on the same line should be formatted so that language statements appear before comments.

Options available with error are:

- -n Do not touch any files; all error messages are sent to the standard output.
- -q The user is queried whether s/he wants to touch the file. A "y" or "n" to the question is necessary to continue. Absence of the -q option implies that all referenced files (except those referring to discarded error messages) are to be touched.
- -v After all files have been touched, overlay the visual editor vi with it set up to edit all files touched, and positioned in the first touched file at the first error. If vi can't be found, try ex or ed from standard places.
- -t Take the following argument as a suffix list. Files whose suffixes do not appear in the suffix list are not touched. The suffix list is dot separated, and "*" wildcards work. Thus the suffix list:

```
".c.y.foo *.h"
```

allows error to touch files ending with ".c", ".v", ".foo*" and ".h".

-s Print out statistics regarding the error categorization. Not too useful.

Error catches interrupt and terminate signals, and if in the insertion phase, will orderly terminate what it is doing.

FILES

~/.errorrc /dev/tty function names to ignore for *lint* error messages user's teletype

BUGS

Opens the teletype directly to do user querying.

Source files with links make a new copy of the file with only one link to it.

Changing a language processor's format of error messages may cause error to not understand the error message.

ERROR(1) ERROR(1)

Error, since it is purely mechanical, will not filter out subsequent errors caused by 'floodgating' initiated by one syntactically trivial error. Humans are still much better at discarding these related errors.

Pascal error messages belong after the lines affected (error puts them before). The alignment of the '|' marking the point of error is also disturbed by error.

Error was designed for work on CRT's at reasonably high speed. It is less pleasant on slow speed terminals, and has never been used on hardcopy terminals.

Error messages referring to the same source line are not guaranteed to be inserted in the same order in which they are produced by the compiler.

NAME

f77 - FORTRAN 77 compiler

SYNOPSIS

f77 [option] ... file ...

DESCRIPTION

NOTE: F77 is at the 4.2BSD level. It supports the 4.2BSD level run time functions. See intro(3f) for lists of supported 4.2BSD functions and unsupported 4.3BSD functions.

F77 is the FORTRAN 77 compiler. It accepts several types of arguments:

Arguments with names ending in .f are taken to be FORTRAN 77 source programs; they are compiled, and each object program is left on the file in the current directory having a name that of the source, with .o substituted for .f.

Arguments with names ending in .F are also taken to be FORTRAN 77 source programs; these are processed by the C preprocessor before being compiled by f77.

Arguments with names ending in .r or .e are taken to be Ratfor or EFL source programs respectively; these are first transformed by the appropriate preprocessor, then compiled by f77.

Arguments with names ending in .c or .s are taken to be C or assembly source programs and are compiled or assembled, producing a .o file.

The following options have the same meaning as in cc(1). See ld(1) for load-time options.

- -c Suppress loading and produce .o files for each source file.
- -g Have the compiler produce additional symbol table information for dbx(1). Also pass the -lg flag to ld(1).
- -o output

Name the final output file output instead of a.out.

- $-\mathbf{p}$ Prepare object files for profiling, see prof(1).
- -pg Causes the compiler to produce counting code in the manner of -p, but invokes a runtime recording mechanism that keeps more extensive statistics and produces a gmon.out file at normal termination. An execution profile can then be generated by use of gprof(1).
- -w Suppress all warning messages. If the option is "-w66", only FORTRAN 66 compatibility warnings are suppressed.
- -**D**name = def
- Dname

Define the *name* to the C preprocessor, as if by "#define". If no definition is given, the name is defined as 1. (.F suffix files only).

- -Idir "#include" files with names not beginning in "/" are always sought first in the directory of the file argument, then in directories named in -I options, then in directories on a standard list. (.F suffix files only).
- −O Invoke an object-code optimizer.
- -S Compile the named programs, and leave the assembler-language output on corresponding files suffixed .s. (No .o is created.).

The following options are peculiar to f77.

- -i2 On machines which support short integers, make the default integer constants and variables short. (-i4 is the standard value of this option). All logical quantities will be short.
- -m Apply the M4 preprocessor to each .r file before transforming it with the Ratfor or EFL preprocessor.

- onetrip

Compile DO loops that are performed at least once if reached. (FORTRAN 77 DO loops are not performed at all if the upper limit is smaller than the lower limit.)

- -u Make the default type of variables "undefined" rather than using the default FORTRAN rules.
- -v Print the version number of the compiler, and the name of each pass as it executes.
- -C Compile code to check that subscripts are within declared array bounds.
- -F Apply the C, EFL, or Ratfor preprocessors to relevant files, put the result in the file with the suffix changed to .f, but do not compile.
- $-\mathbf{E}x$ Use the string x as an EFL option in processing e files.
- $-\mathbf{R}x$ Use the string x as a Ratfor option in processing r files.

-N[qxscn]nnn

Make static tables in the compiler bigger. The compiler will complain if it overflows its tables and suggest you apply one or more of these flags. These flags have the following meanings:

- q Maximum number of equivalenced variables. Default is 150.
- x Maximum number of external names (common block names, subroutine and function names). Default is 200.
- s Maximum number of statement numbers. Default is 401.
- c Maximum depth of nesting for control statements (e.g. DO loops). Default is 20.
- n Maximum number of identifiers. Default is 1009.
- -U Do not convert upper case letters to lower case. The default is to convert FORTRAN programs to lower case except within character string constants.

Other arguments are taken to be either loader option arguments, or F77-compatible object programs, typically produced by an earlier run, or perhaps libraries of F77-compatible routines. These programs, together with the results of any compilations specified, are loaded (in the order given) to produce an executable program with name a.out.

FILES

file.[fFresc]	input file
file.o	object file
a.out	loaded output
/usr/lib/f77pass1	compiler
/lib/f1	pass 2
/lib/c2	optional optimizer
/lib/cpp	C preprocessor
/usr/lib/libF77.a	intrinsic function library
/usr/lib/libI77.a	FORTRAN I/O library
/usr/lib/libU77.a	interface library
/usr/lib/libF77_p.a	profiling intrinsic function library
/usr/lib/libI77_p.a	profiling FORTRAN I/O library
/usr/lib/libU77_p.a	profiling interface library
/lib/libc.a	C library, see Section 3
/lib/libc_p.a	profiling C library
mon.out	file produced for analysis by prof(1).
gmon.out	file produced for analysis by gprof(1).

SEE ALSO

- S. I. Feldman, P. J. Weinberger, A Portable FORTRAN 77 Compiler
- D. L. Wasley, Introduction to the f77 I/O Library
- cc(1), efl(1), gprof(1), ld(1), prof(1), ratfor(1)

DIAGNOSTICS

The diagnostics produced by f77 itself are intended to be self-explanatory. Occasional messages may be produced by the loader.

BUGS

Floating point expressions are evaluated using IEEE 754 double-precision arithmetic with default rounding and exception handling. There is presently no support for setting or interrogating rounding and exception modes or exception flags.

)

F77(1)

NAME

hc - High C compiler

SYNOPSIS

hc [option] ... file ...

DESCRIPTION

Hc accepts several types of arguments:

Arguments with names ending in ".c" are taken to be C source programs; they are compiled, and each object program is left with a corresponding file name ending in ".o". The ".o" file is normally deleted, however, if a single C program is compiled and loaded at one time.

In the same way, arguments with names ending in ".s" are taken to be assembly source programs and are assembled, producing a ".o" file.

The following options are interpreted by hc. See ld(1) for load-time options. (By convention, options applicable only to High C are prefixed with an H.)

- -c Suppress the loading phase of the compilation, and force an object file to be produced even if only one program is compiled.
- Have the compiler produce additional symbol-table information for dbx(1). Also pass the $-\lg$ flag to ld(1). This flag turns off certain optimizations that complicate debugging by rearranging code. Use the -O flag in conjunction with -g if you wish all normal optimizations to be included.
- -w Suppress warning diagnostics.
- Arrange for the compiler to produce code that counts the number of times each routine is called during execution. If loading takes place, search the profiling library /usr/lib/libc_p.a in lieu of the standard C library /lib/libc.a. Also replace the standard startup routine by one that automatically calls monitor(3) at the start and arranges to write out a mon.out file at normal termination of execution of the object program. An execution profile can then be generated by use of prof(1).
- -pg Like -p, but invoke a run-time recording mechanism that keeps more extensive statistics and produces a *gmon.out* file at normal termination. An execution profile can then be generated by use of gprof(1).
- Perform all optimizations supported by the compiler. This is the default unless $-\mathbf{g}$ is specified. Therefore, this option has meaning only when used in conjunction with $-\mathbf{g}$.
- -R Pass this flag on to as, making initialized variables shared and read-only.
- -S Compile the named C programs, and leave the assembler-language output on corresponding files suffixed ".s". Hc normally generates an object module directly, i.e. without producing an intermediate assembler source file. If the Hasm flag is also specified, the ".s" file is annotated.
- -M Run only the cpp macro preprocessor on the named C programs, requesting it to generate Makefile dependencies and send the result to the standard output.
- -E Run only the cpp macro preprocessor on the named C programs, and send the result to the standard output.
- -C Prevent the cpp macro preprocessor from removing comments.

-o output

Name the final output file *output*. If this option is used, the file **a.out** will be left undisturbed.

- -Dname = def
- Dname

Define name to the preprocessor, as if by "#define". If no definition is given, name is defined as 1 (one).

- Uname

Remove any initial definition of name.

- -Idir "#include" files that do not have names beginning in "/" are always sought first in the directory of the file argument, then in directories named in -I options, and finally in directories on a standard list.
- -Ldir Library archives are sought first in directories named in -L options, then in directories on a standard list.

- Bstring

Find substitute compiler executables in the files named string with the suffixes "cpp" and "hccom." If string is empty, it defaults to |usr/c|o.

- -v Display each subprocess being executed on standard error.
- -mx Compile using some machine-dependent options. Currently available options are:
 - -ma Support use of alloca(3) by the compiled function. Alloca places some constraints on compiled code, making it slightly less efficient. Only the individual functions that call alloca need be compiled with -ma.
 - -ms Forces the compiler to put out minimum-size floating point data blocks. (Normally they are generously padded.) This guarantees that the size of objects will remain approximate to that of previous releases, at a small cost in performance. See "IBM/4.3 Linkage Convention" in Volume II, Supplementary Documents, for more information.

- Hasm

Produce a pseudo-assembler listing of the generated code on standard output, annotated with lines from the main source file. These lines appear as comments immediately preceding the corresponding assembly instructions. If the $-\mathbf{S}$ option is also specified, the generated ".s" file is annotated with lines from the source file and no listing is written to standard output.

— Нсрр

Invoke the cpp macro preprocessor on the source file prior to compiling. This is the default.

- Hnocpp

Use hc's ANSI-conforming macro preprocessor instead of cpp.

- Hlines = n

Cause a page-eject to occur after every n lines written to standard output. The default is 60. This option is used in conjunction with the -Hlist and -Hasm options. If n is 0, no page-ejects are emitted.

- Hlist Produce a source listing on standard output.
- Hon = toggle
- Hoff = toggle

Turn toggle on or off. (See Compiler Toggles in the "High C Programmer's Guide" for a description of the compiler toggles.)

- Hppo
- Hppo = file

Run only hc's ANSI-conforming macro preprocessor. The result is sent to standard

output or file, if specified. This flag is analogous to $-\mathbf{E}$.

- Hvolatile

Force compiler to read from memory on all pointer dereferences. This is necessary only when pointers are used as addresses whose contents are "volatile" (can change via external forces).

-H+w

Issue all warnings.

!...... C1...

Other arguments are taken to be either loader option arguments or C-compatible object programs typically produced by an earlier hc or pcc run, or perhaps libraries of C-compatible routines. These programs, together with the results of any compilations specified, are loaded (in the order given) to produce an executable file named a.out.

FILES

input file
object file
loaded output
temporary
outboard macro preprocessor
compiler
backup compiler
backup preprocessor
runtime startoff
startoff for profiling
startoff for gprof-profiling
standard library, see intro(3)
profiling library, see intro(3)
directory for #include files
file produced for analysis by prof(1)
file produced for analysis by gprof(1)

SEE ALSO

- B. W. Kernighan and D. M. Ritchie, The C Programming Language, Prentice-Hall, 1978
- B. W. Kernighan, Programming in C-a tutorial
- D. M. Ritchie, C Reference Manual
- "IBM/4.3 Linkage Convention" in Volume II, Supplementary Documents
- "High C Programmer's Guide" in Appendix C
- High C Language Extensions Manual with Rationale and Tutorials, from MetaWare Incorporated.
- "Recompiling with High C" in Volume II, Supplementary Documents
- adb(1), as(1), cc(1), dbx(1), gprof(1), ld(1), pcc(1), prof(1), malloc(3), monitor(3)

DIAGNOSTICS

The diagnostics produced by hc are intended to be self-explanatory. Occasional messages may be produced by the assembler or loader.

KBDLOCK(1) KBDLOCK(1)

NAME

kbdlock - lock the keyboard of the IBM RT PC

SYNOPSIS

kbdlock [bootdev] [off]

DESCRIPTION

Kbdlock turns off console keyboard scanning and forces the system to boot from bootdev if rebooted. The keyboard scanning continues to be disabled over a reboot. Legal values for bootdev are {hd,fd}[0-3], hd for hard disks, fd for diskette drives. The default is hd0. Once kbdlock is executed, the console keyboard scanning can be enabled by either turning the keylock on the front cover off and on, or executing the command kbdlock off. This re-enables keyboard scanning and restores the boot order to its original condition.

Kbdlock does not affect the operations of the mouse, speaker or serial lines, or the ability of the system to reboot.

Kbdlock is a software alternative to physically locking the machine. It allows the machine to be rebooted; physical locking does not allow rebooting.

NOTES

Kbdlock is not supported on the IBM 6152 Academic System.

SEE ALSO

kbdemul(4)

NAME

ld - link editor

SYNOPSIS

ld [option] ... file ...

DESCRIPTION

Ld combines several object programs into one, resolves external references, and searches libraries. In the simplest case, several object files are given, and ld combines them, producing an object module that can either be executed or become the input for a further ld run. (In the latter case, the -r option must be given to preserve the relocation bits.) The output of ld is normally placed in a.out. This file is made executable only if no errors occurred during the load.

The argument routines are concatenated in the order specified. The entry point of the output is the beginning of the first routine (unless the -e option is specified).

If any argument is a library, it is searched exactly once at the point it is encountered in the argument list. Only those routines defining an unresolved external reference are loaded. If a routine from a library references another routine in the library, and the library has not been processed by ranlib(1), the referenced routine must appear after the referencing routine in the library. Thus the order of programs within libraries may be important. The first member of a library should be a file named "__SYMDEF", which is understood to be a dictionary for the library as produced by ranlib(1); the dictionary is searched iteratively to satisfy as many references as possible.

The symbols "_etext", "_edata" and "_end" ("etext", "edata" and "end" in C) are reserved and, if referenced, set to the first location above the program, the first location above initialized data, and the first location above all data respectively. It is erroneous to define these symbols.

Ld looks for ".oVncs" in the symbol table of each file, and prints an error report if it is present in some, but not in all, files. (This is a transition aid for installations with pre-release .o files.)

Ld may use trampoline code when the object module contains more than 2²⁰ bytes of text. See "4.3/RT Linkage Convention".

Ld understands several options. Except for -1, they should appear before the file names.

- This option specifies incremental loading; i.e. linking is to be done in a manner so that the resulting object may be read into an already executable program. The next argument is the name of a file whose symbol table will be taken as a basis on which to define additional symbols. Only newly linked material will be entered into the text and data portions of a.out, but the new symbol table will reflect every symbol defined before and after the incremental load. This argument must appear before any other object file in the argument list. The -T option may be used as well, and will be taken to mean that the newly linked segment will begin at the corresponding address (which must be a multiple of pagesize(1)). The default value is the old value of _end.
- -D Take the next argument as a hexadecimal number and pad the data segment with zero bytes to the stated length.
- -d Force definition of common storage even if the -r flag is present.
- -e The following argument is taken to be the symbolic name of the entry point of the loaded program; location 0 is the default.
- -K Relocate the data segment of -n and -z formats relative to 0; otherwise, it will be relocated to 0x10000000. This is used when linking the kernel.
- -Ldir Add dir to the list of directories in which libraries are searched for. Directories specified with -L are searched before the standard directories.
- -lx This option is an abbreviation for a library named /lib/libx.a, /usr/lib/libx.a, or /usr/local/lib/libx.a, whichever is found first. A library is searched when its name is

1

- encountered, so the placement of a -1 is significant.
- -M Produce a primitive load map, listing the names of the files to be loaded. A -MM option indicates, along with each name listed, whether either of the symbols ".oVocs" or ".oVncs" was found in the file, and suppresses the mixed ocs/ncs diagnostic.
- -N Do not make the text portion read-only or sharable. (Use "magic number" 0407. Magic numbers are explained in a.out(5).)
- -n Arrange (by giving the output file a 0410 "magic number") that when the output file is executed, the text portion will be read-only and shared among all users executing the file. This involves moving the data areas to 0x10000000.
- $-\mathbf{o}$ The name argument after $-\mathbf{o}$ is used as the name of the *ld* output file, instead of **a.out**.
- -r Generate relocation bits in the output file so that it can be the subject of another *ld* run. This flag also prevents final definitions from being given to common symbols, and suppresses the **undefined symbol** diagnostics.
- -S Strip the output by removing all symbols except locals and globals.
- -s Strip the output, that is, remove the symbol table and relocation bits to save space (but impair the usefulness of the debuggers). This information can also be removed by strip(1).
- -T The next argument is a hexadecimal number that sets the text segment origin. The default origin is 0.
- -t Trace. Print the name of each file as it is processed.
- -u Take the following argument as a symbol and enter it as undefined in the symbol table. This is useful for loading wholly from a library, since initially the symbol table is empty and an unresolved reference is needed to force the loading of the first routine.
- -X Save local symbols except for those having names beginning with "L".
- -x Do not preserve local (non-.globl) symbols in the output symbol table; enter only external symbols. This option saves some space in the output file.
- -ysym Show each file in which sym appears, its type and whether the file defines or references it. Many such options may be given to trace many symbols. (It is usually necessary to begin sym with "_", as external C, FORTRAN and Pascal variables begin with underscores.)
- Arrange for the process to be loaded on demand from the resulting executable file (0413 format) rather than preloaded. This is the default; it results in a pagesize(1)-byte header on the output file followed by a text and data segment, the size of each being a multiple of pagesize bytes (being padded out with nulls in the file if necessary). With this format, the first few BSS segment symbols may actually appear (from the output of size(1)) to live in the data segment; this avoids wasting the space resulting from data segment size roundup. Ld places the first byte of data at 0x100000000.

FILES

```
/lib/lib*.a libraries
/usr/lib/lib*.a more libraries
/usr/local/lib/lib*.a still more libraries
a.out output file
```

SEE ALSO

```
as(1), ar(1), cc(1), ranlib(1)
"4.3/RT Linkage Convention" in Volume II, Supplementary Documents
```

LD(1) LD(1)

DIAGNOSTICS

file.o: Mixed use of old and new calling sequences.

file.o: No NCS flag present. The symbol table of the indicated file does not contain the symbol ".oVncs".

BUGS

There is no way to force data to be page-aligned.

LEARN(1) LEARN(1)

NAME

learn - computer aided instruction about UNIX

SYNOPSIS

```
learn [ - directory ] [ subject [ lesson ] ]
```

DESCRIPTION

Learn gives Computer Aided Instruction courses and practice in the use of the UNIX operating system, the C Shell, and the Berkeley text editors. To get started, simply type learn. If you used learn before and left your last session without completing a subject, the program will use information in \$HOME/.learnrc to start you up in the same place you left off. Your first time through, learn asks questions to find out what you want to do. Some questions may be bypassed by naming a subject, and more yet by naming a lesson. You may enter the lesson as a number that learn gave you in a previous session. If you do not know the lesson number, you may enter the lesson as a word, and learn will look for the first lesson containing it. If the lesson is "-", learn prompts for each lesson; this is useful for debugging.

The subjects presently handled are

files editor vi morefiles macros eqn C

There are a few special commands. The command "bye" terminates a *learn* session and "where" tells you of your progress, with "where m" telling you more. The command "again" re-displays the text of the lesson and "again *lesson*" lets you review *lesson*. There is no way for *learn* to tell you the answers it expects in English, however, the command "hint" prints the last part of the lesson script used to evaluate a response, while "hint m" prints the whole lesson script. This is useful for debugging lessons and might possibly give you an idea about what it expects.

The - directory option allows one to exercise a script in a nonstandard place.

FILES

/usr/lib/learn subtree for all dependent directories and files /usr/tmp/pl* playpen directories startup information

SEE ALSO

```
csh(1), ex(1)
B. W. Kernighan and M. E. Lesk, LEARN - Computer-Aided Instruction on UNIX
```

BUGS

The main strength of *learn*, that it asks the student to use the operating system while learning, also makes possible baffling mistakes. It is helpful, especially for nonprogrammers, to have an experienced user near at hand during the first sessions.

Occasionally lessons are incorrect, sometimes because the local version of a command operates in a non-standard way. Occasionally a lesson script does not recognize all the different correct responses, in which case the "hint" command may be useful. Such lessons may be skipped with the "skip" command, but it takes some sophistication to recognize the situation.

To find a *lesson* given as a word, *learn* does a simple fgrep(1) through the lessons. It is unclear whether this sort of subject-indexing is better than none.

Spawning a new shell is required for each of many user and internal functions.

MSET(1) MSET(1)

NAME

mset - retrieve ASCII to IBM 3270 keyboard map

SYNOPSIS

mset [-picky] [-shell] [keyboardname]

DESCRIPTION

Mset retrieves mapping information for the ASCII keyboard to IBM 3270 terminal special functions. Normally, these mappings are found in /etc/map3270 (see map3270(5)). This information is used by the tn3270 command (see tn3270(1)).

The default *mset* output can be used to store the mapping information in the process environment in order to avoid scanning /etc/map3270 each time tn3270 is invoked. To do this, place the following command in your .login file:

set noglob; setenv MAP3270 "'mset'"; unset noglob

If the keyboardname argument is not supplied, mset attempts to determine the name of the keyboard the user is using, by checking the KEYBD environment variable. If the KEYBD environment variable is not set, then mset uses the user's terminal type from the environment variable TERM as the keyboard name. Normally, mset then uses the file /etc/map3270 to find the keyboard mapping for that terminal. However, if the environment variable MAP3270 exists and contains the entry for the specified keyboard, then that definition is used. If the value of MAP3270 begins with a slash ('/') then it is assumed to be the full pathname of an alternate mapping file and that file is searched first. In any case, if the mapping for the keyboard is not found in the environment, nor in an alternate map file, nor in the standard map file, then the same search is performed for an entry for a keyboard with the name unknown. If that search also fails, then a default mapping is used.

The arguments to *mset* are:

-picky When processing the various map 3270 entries (for the user's keyboard, and all those encountered before the one for the user's keyboard), mset normally will not complain about entries for unknown functions (like "PFX1"); the -picky argument causes mset to issue warning messages about these unknown entries.

-shell If the map3270 entry is longer than the shell's 1024 environmental variable length limit, the default *mset* output cannot be used to store the mapping information in the process environment to avoid scanning /etc/map3270 each time tn3270 is invoked. The -shell argument causes mset to generate shell commands to set the environmental variables MAP3270, MAP3270A, and so on, breaking up the entry to fit within the shell environmental variable length limit. To set these variables, place the following command in your .login file:

mset -shell > tmp; source tmp; /bin/rm tmp

keyboardname

When searching for the map3270 entry that matches the user's keyboard, mset will use keyboardname instead of determining the keyboard name from the KEYBD or TERM environmental variables.

FILES

/etc/map3270 keyboard mapping for known keyboards

SEE ALSO

tn3270(1), map3270(5)

NAME

mt - magnetic tape manipulating program

SYNOPSIS

mt [-f tapename | command [count]

DESCRIPTION

Mt is used to give commands to a magnetic tape drive. If a tape name is not specified, the environment variable TAPE is used; if TAPE does not exist, mt uses the device |dev|rmt12. Note that tapename must reference a raw (not block) tape device. By default mt performs the requested operation once. Operations may be performed multiple times by specifying count.

The available commands are listed below. Only as many characters as are required to uniquely identify a command need be specified.

eof, weof

Write count end-of-file marks at the current position on the tape.

fsf Forward space count files.

fsr Forward space count records.

bsf Back space count files.

bsr Back space count records.

rewind Rewind the tape. (Count is ignored.)

offline, rewoffl

Rewind the tape and place the tape unit off-line. (Count is ignored.)

status Print status information about the tape unit.

retension

Retension the tape by spacing forward to the end and then rewinding. It is recommended that new tape be retensioned before use. (Count is ignored.)

erase Erase the entire tape. This automatically retensions it. (Count is ignored.)

Mt returns a 0 exit status if an operation was successful, 1 if a command was unrecognized, and 2 if an operation failed.

FILES

/dev/nrst0 Raw streaming tape interface with no rewind

/dev/rst0

Raw streaming tape interface with built-in rewind

SEE ALSO

dd(1), ioctl(2), mtio(4), environ(7), st(4)

BUGS

The streaming tape hardware can back up, but only at a snail's pace. For this reason, bsf is not implemented, and bsr should be used only with very small counts.

The default device (/dev/rmt/2) is inappropriate for the RT PC.

NAME

pcc - pcc-based C compiler

SYNOPSIS

pcc [option] ... file ...

DESCRIPTION

Pcc accepts several types of arguments:

Arguments with names ending in ".c" are taken to be C source programs; they are compiled, and each object program is left with a corresponding file name ending in ".o". The ".o" file is normally deleted, however, if a single C program is compiled and loaded at one time.

In the same way, arguments with names ending in ".s" are taken to be assembly source programs and are assembled, producing a ".o" file.

The following options are interpreted by pcc. See ld(1) for load-time options.

- -c Suppress the loading phase of the compilation, and force an object file to be produced even if only one program is compiled.
- -g Have the compiler produce additional symbol-table information for dbx(1). Also pass the -lg flag to ld(1).
- -w Suppress warning diagnostics.
- Arrange for the compiler to produce code that counts the number of times each routine is called during execution. If loading takes place, search the profiling library /usr/lib/libc_p.a in lieu of the standard C library /lib/libc.a. Also replace the standard startup routine by one that automatically calls monitor(3) at the start and arranges to write out a mon.out file at normal termination of execution of the object program. An execution profile can then be generated by use of prof(1).
- -pg Like -p, but invoke a run-time recording mechanism that keeps more extensive statistics and produces a *gmon.out* file at normal termination. An execution profile can then be generated by use of *gprof*(1).
- -O Invoke an object-code improver. The following flags specify that only certain optimizations are to be performed:
 - -Ob Do balr optimizations.
 - Oj Do jump optimizations.
 - -OI Do load optimizations.
 - -On When used with the -S flag, compiler output containing optimizer directives is produced; otherwise it specifies that no optimizations are to be performed.
 - Or Do register variable optimizations.
 - -Om Do move register optimizations.
 - -Ox Do miscellaneous optimizations.

The following flags may be specified for debugging the optimizer.

- -Ot Turn on tracing.
- Od Turn on debugging.
- -Os Show modifications resulting from optimization in the output file.
- -R Pass this flag on to as, making initialized variables shared and read-only.
- -S Compile the named C programs, and leave the assembler-language output on corresponding files suffixed ".s".

- -M Run only the macro preprocessor on the named C programs, requesting it to generate Makefile dependencies and send the result to the standard output.
- -E Run only the macro preprocessor on the named C programs, and send the result to the standard output.
- -C prevent the macro preprocessor from removing comments.

-o output

Name the final output file *output*. If this option is used, the file **a.out** will be left undisturbed.

- -Dname = def
- Dname

Define name to the preprocessor, as if by "#define". If no definition is given, name is defined as 1 (one).

- Uname

Remove any initial definition of name.

- -Idir "#include" files that do not have names beginning in "/" are always sought first in the directory of the *file* argument, then in directories named in -I options, and finally in directories on a standard list.
- -Ldir Library archives are sought first in directories named in -L options, then in directories on a standard list.

- Bstring

Find substitute compiler passes in the files named string with the suffixes "cpp," "ccom," and "c2". If string is empty, it defaults to /usr/c/o.

-t[p012]

Find only the designated compiler passes in the files whose names are constructed by a $-\mathbf{B}$ option. In the absence of a $-\mathbf{B}$ option, the *string* is taken to be |usr/c|.

- -v Pcc debugging flag. Displays the programs being called and their arguments on standard error.
- -mx Compile using some machine-dependent options. Currently available options are:
 - -ma Support use of alloca(3) by the compiled function. Alloca places some constraints on compiled code, making it slightly less efficient. Only the individual functions that call alloca need be compiled with -ma.
 - -ms Forces the compiler to put out minimum-size floating point data blocks. (Normally they are generously padded.) This guarantees that the size of objects will remain approximate to that of previous releases, at the expense of performance. See "IBM/4.3 Linkage Convention" in Volume II, Supplementary Documents, for more information.

Several compiler debugging flags are also available. A flag may be repeated, to obtain more voluminous output. Usually repetitions beyond the fifth have no effect.

- -xl Include the source line numbers as comments in the assembler output.
- -xd Ddebug flag for debugging the first pass of the compiler.
- -xi Idebug flag for debugging the first pass of the compiler.
- -xb Bdebug flag for debugging the first pass of the compiler.
- -xt Tdebug flag for debugging the first pass of the compiler.

- -xe Edebug flag for debugging the first pass of the compiler.
- -xx Xdebug flag for debugging the first pass of the compiler.
- -xp Expression flag for debugging the second pass of the compiler.
- -xo Orders flag for debugging the second pass of the compiler.
- -xr Register allocation flag for debugging the second pass of the compiler.
- -xa Rallo flag for debugging the second pass of the compiler.
- -xv Vdebug flag for debugging the second pass of the compiler.
- -xc Ttype calls flag for debugging the second pass of the compiler.
- -xs Shapes flag for debugging the second pass of the compiler.
- xu Sethi-Ullman flag for debugging the second pass of the compiler.
- -xm General machine-dependent flag for debugging the second pass of the compiler.

The flags -f and -Hxxx are not supported (-Hxxx is supported by hc(1) only); specifying these flags will cause a warning message, though compilation will continue.

Other arguments are taken to be either loader option arguments or C-compatible object programs typically produced by an earlier *pcc* or *hc* run, or perhaps libraries of C-compatible routines. These programs, together with the results of any compilations specified, are loaded (in the order given) to produce an executable file named **a.out**.

FILES

```
file.c
                 input file
file.o
                 object file
a.out
                 loaded output
/tmp/ctm?
                 temporary
/lib/cpp
                 preprocessor
/lib/ccom
                 compiler
/usr/c/occom
                 backup compiler
                 backup preprocessor
/usr/c/ocpp
/lib/c2
                 optional optimizer
/lib/crt0.o
                 runtime startoff
                 startoff for profiling
/lib/mcrt0.o
/usr/lib/gcrt0.o
                 startoff for gprof-profiling
/lib/libc.a
                 standard library, see intro(3)
/usr/lib/libc p.a profiling library, see intro(3)
/usr/include
                 directory for #include files
mon.out
                 file produced for analysis by prof(1)
gmon.out
                 file produced for analysis by gprof(1)
```

SEE ALSO

- B. W. Kernighan and D. M. Ritchie, The C Programming Language, Prentice-Hall, 1978
- B. W. Kernighan, Programming in C-a tutorial
- D. M. Ritchie, C Reference Manual
- "IBM/4.3 Linkage Conventions" in Volume II, Supplementary Documents adb(1), as(1), cc(1), dbx(1), gprof(1), hc(1), ld(1), prof(1), malloc(3), monitor(3)

DIAGNOSTICS

The diagnostics produced by pcc are intended to be self-explanatory. Occasional messages may be produced by the assembler or loader.

BUGS

The load, register-variable and balr optimizations contain bugs.

PCC(1)

Currently, only jump and move-register optimizations are enabled.

Certain simple expressions, particularly constant expressions used as initializers, may be reported to be too complex. Rewrite the expression, or decompose it into two expressions.

The value resulting from an assignment to a *char* or *short* is not truncated to the width of the *lvalue*. This affects only assignment subexpressions contained within larger expressions. Thus:

$$int3 = char2 = -1$$

sets char2 to 255 and int3 to -1.

Multiple assignment of structs or unions causes a compiler error. Rewrite:

$$struct3 = struct2 = struct1$$

as two assignment statements.

Frame size (space occupied by auto variables) is limited to 32K.

Hc(1) has none of these restrictions; consider using it as an alternative to pcc.

NAME

pf - set keyboard program-function keys

SYNOPSIS

$$pf[-f codes][-d][-S][-z][-r][-e command][file...]$$

DESCRIPTION

Pf redefines keys on the IBM RT PC or IBM 6152 Academic System keyboard. It can be called interactively or it can be invoked from a script, such as from .login or .profile during login processing. Options and files can be interspersed on the command line and are processed in the order encountered. If no files or -e options are specified, input is from standard input. The following options are available:

- -e Take the next argument as a pf command. This is the usual way of doing quick definitions from the keyboard or a shell script.
- -d Print debugging messages.
- -f Take the following argument as the codes file instead of the default /usr/lib/keyboard_codes (see keyboard_codes(5)). This file contains the mapping of scan codes to key names.
- -S Replace the standard set of keyboard definitions with the current set (see *kbdemul*(4)). This can be done only by the super-user.
- -r Reset the current keyboard definitions to the standard definitions.
- -z Clear the current keyboard string definitions. This is useful for clearing out old definitions to make room for a new set.

COMMANDS

The following commands can be issued interactively or from a file (e.g., during .login processing). Pf ignores empty lines and lines that begin with #.

help Print a short help message.

display name print name

Display the current definitions associated with name. Name is either a single scan code or the word all. A scan code consists of an optional prefix followed by a scan-number (decimal) or by the symbol on the appropriate keycap. The prefixes are shift-, control-(or various synonyms), alt-, or action-. For example, alt-a specifies the alt shift of the a key.

set name = string

Specify a new definition for name. String may use "\" and "^" to specify special characters that would otherwise be difficult to enter (or would take effect immediately). The "\" can be followed by the standard C character escapes. The "^" specifies a control character. String is terminated by the end of the line.

reset Reset the keyboard definitions to the standard definitions.

sstd Set the standard definitions from the current definitions. This may only be done by the super-user.

safe on safe yes

Turn on safe mode. In safe mode, it is not possible to redefine normal alphabetic characters and the < Enter > key.

safe off

safe no Turn off safe mode. In unsafe mode, it is possible to redefine any key on the keyboard.

click off

click no

Turn off all keyboard clicks (silent keystrokes). Not supported for the IBM 6152 Academic System.

click hard

click on

click yes

Turn on hardware keyboard clicks (default). The hardware click is a short, sharp click. Not supported for the IBM 6152 Academic System.

click soft

Turn on software keyboard clicks. The software click is entirely independent of the hardware click. It has a different tone and duration. Not supported for the IBM 6152 Academic System.

click both

Turn on both hardware and software clicks. Not supported for the IBM 6152 Academic System.

click? Print the current keyboard click mode. Not supported for the IBM 6152 Academic System.

end Terminate pf.

quit Terminate pf.

NOTES

It is possible to redefine a key to have a *meta* or *shift* function. In this case, *string* is the name of the *meta* function enclosed in braces. For example, the command line:

set alt-page-up = {FN SWITCH}

causes the < Alt > key plus the < Page-Up > key to switch the console output to the next screen (the FN_SWITCH function). For more details on the available functions, see *kbdemul(4)*. If it is necessary to generate a string that is the name of a function key inside braces, then the backslash octal escape for the left brace (\173) can be used.

Pf defaults to safe mode when invoked with input from a terminal; otherwise it operates in unsafe mode. To override this, issue the command safe off.

It is possible to use pf to reorganize the keyboard into different key layouts; in this case, move the key caps and change the /usr/lib/keyboard codes file.

Pf can only be used if kbdemul(4) is the input emulator.

EXAMPLES

The following pf command defines the Alt shift of the b key to generate a Ctrl-Z (" Z ") and then issue the "bg" csh command, to put the currently-running process in the background:

set alt-b =
$$^Zbg^M$$

The following remaps keys at each login. In .login, place the line:

and create the file .login.kbd:

```
# reset to standard, put caps-lock
# and control in convenient places,
# define control-' and alt-b.
reset
```

```
set caps-lock = {FN_CONTROL}
set ctrl = {FN_CAPS_LOCK}
set control-'= ^M^Z^M
set alt-b = ^Zbg^M
click off
```

To turn key click off:

```
pf -e "click off"
```

Click off is not supported for the IBM 6152 Academic System.

FILES

/usr/lib/keyboard_codes defines the standard scancode bindings.

DIAGNOSTICS

Diagnostics are intended to be self-explanatory, except for messages of the form:

```
ioctl KBD type reason
```

which indicate the given *ioctl* failed for the given reason.

SEE ALSO

cons(4), kbdemul(4), keyboard_codes(5)

BUGS

Pf does not reorganize the string definitions when the string area is full.

PIC(1)

NAME

pic - troff preprocessor for drawing simple pictures

SYNOPSIS

pic [-Tt][files]

DESCRIPTION

Pic is a troff(1) preprocessor for drawing simple figures on a typesetter. The basic objects are box, line, arrow, circle, ellipse, arc, and text.

The optional argument -Tt specifies typesetter t; currently supported typesetters are 3812 (the default for the IBM 3812 Pageprinter).

SEE ALSO

B.W. Kernighan, PIC - A Graphics Language for Typesetting

NAME

pp - Professional Pascal compiler

SYNOPSIS

pp [option] ... file ...

DESCRIPTION

Pp accepts several types of arguments:

Arguments with names ending in ".p" are taken to be Pascal source programs; they are compiled, and each object program is left with a corresponding file name ending in ".o". The ".o" file is normally deleted, however, if a single Pascal program is compiled and loaded at one time.

In the same way, arguments with names ending in ".s" are taken to be assembly source programs and are assembled, producing ".o" files.

Also, arguments ending in ".c" are taken to be C source modules and are compiled with High C. (See hc(1).)

The following options are interpreted by pp. See ld(1) for load-time options. (By convention, options applicable only to Professional Pascal and High C are prefixed with an H.)

- -c Suppress the loading phase of the compilation, and force an object file to be produced even if only one program is compiled.
- Have the compiler produce additional symbol-table information for dbx(1). Also pass the $-\lg$ flag to ld(1). This flag turns off certain optimizations that complicate debugging by rearranging code. Use the $-\mathbf{O}$ flag in conjunction with $-\mathbf{g}$ if you wish all normal optimization to be included.
- -w Suppress warning diagnostics.
- Arrange for the compiler to produce code that counts the number of times each routine is called during execution. If loading takes place, search the profiling library /usr/lib/libc_p.a in lieu of the standard C library /lib/libc.a. Also replace the standard startup routine by one that automatically calls monitor(3) at the start and arranges to write out a mon.out file at normal termination of execution of the object program. An execution profile can then be generated by use of prof(1).
- -pg Like -p, but invoke a run-time recording mechanism that keeps more extensive statistics and produces a *gmon.out* file at normal termination. An execution profile can then be generated by use of *gprof*(1).
- Perform all optimizations supported by the compiler. This is the default unless $-\mathbf{g}$ is specified. Therefore, this option has meaning only when used in conjunction with $-\mathbf{g}$.
- -S Compile the named Pascal and C programs, and leave the assembler-language output on corresponding files suffixed ".s". *Pp* normally generates an object module directly, i.e. without producing an intermediate assembler source file. If the -Hasm flag is also specified, the ".s" file is annotated.
- -M Run only the macro preprocessor on the named Pascal and C programs, requesting it to generate Makefile dependencies and send the result to the standard output.
- -E Run only the outboard macro preprocessor on the named Pascal and C programs, and send the result to the standard output.
- -C prevent the macro preprocessor from removing comments.
- -o output

Name the final output file *output*. If this option is used, the file **a.out** will be left undisturbed.

- -Dname = def
- Dname

Define name to the preprocessor, as if by "#define". If no definition is given, name is defined as 1.

- Uname

Remove any initial definition of name.

- -Idir "#include" files that do not have names beginning in "/" are always sought first in the directory of the file argument, then in directories named in -I options, and finally in directories on a standard list.
- -Ldir Library archives are sought first in directories named in -L options, then in directories on a standard list.

Bstring

Find substitute compiler passes in the files named string with the suffixes "cpp" and "ppcom".

- -v Displays each subprocess being executed on stderr.
- Hansi Accept only programs conforming to the ANSI Pascal standard.
- Hasm Produce a pseudo-assembler listing of the generated code on standard output, annotated with lines from the main source file. These lines appear as comments immediately preceding the corresponding assembly instructions. If the -S option is also specified, the generated ".s" file is annotated with lines from the source file and no listing is written to standard output.

- Hcheap

Use C heap manager (malloc) instead of the Pascal heap manager.

- Hcpp Invoke the outboard C macro preprocessor on the source files prior to compiling. This is the default.

- Hnocpp

Suppress invocation of the outboard C macro preprocessor.

- Hlines = n

Cause a page-eject to occur after every n lines written to standard output. The default is 60. This option is used in conjunction with the -Hist and -Hasm options. If n is 0, no page-ejects are emitted.

- Hlist Produce a source listing on standard output.
- Hon = toggle
- Hoff = toggle

Turn toggle on or off. See Compiler Toggles in the Programmer's Guide for a description of the compiler toggles.

Other arguments are taken to be either loader option arguments or object programs typically produced by an earlier pp run, or perhaps archive libraries of routines. These programs, together with the results of any compilations specified, are loaded (in the order given) to produce an executable program named a.out.

FILES

file.[pc]	input file
file.o	object file
a.out	loaded output
/tmp/pp.*	temporary
/lib/cpp	preprocessor
/usr/lib/ppcom	compiler

/lib/crt0.o runtime startoff /lib/mcrt0.o startoff for profiling /usr/lib/gcrt0.o startoff for gprof-profiling /lib/libc.a standard library, see intro(3) /usr/lib/libc p.a profiling library, see intro(3) /usr/lib/pp/includeStandard PP include files /usr/lib/pp/libpp.aPascal library /usr/lib/pp/cheap.oHeap allocation /usr/lib/pp/pheap.oHeap allocation /usr/lib/pp/ppansi.stANSI standard scan tables /usr/lib/pp/ppansi.ptANSI standard parse tables directory for #include files /usr/include file produced for analysis by prof(1) mon.out gmon.out file produced for analysis by gprof(1)

SEE ALSO

"IBM/4.3 Linkage Convention" in Volume II, Supplementary Documents MetaWare *Professional Pascal Documentation Set* (one set shipped free with each license), adb(1), as(1), dbx(1), gprof(1), hc(1), ld(1), prof(1), monitor(3)

DIAGNOSTICS

The diagnostics produced by pp itself are intended to be self-explanatory. Occasional messages may be produced by the assembler or loader.

NAME

pprint - print text files on IBM 3812 Pageprinter

SYNOPSIS

pprint [options] [files]

DESCRIPTION

Pprint filters input through pr and sends the resulting output to an IBM 3812 Pageprinter. If no files are specified, pprint reads the standard input.

Pprint accepts a number of options which control job parameters, page layout, font selection, and input filtering. Some flags accept optional arguments which must be immediately adjacent to the corresponding flag. Keywords (printed in boldface, below) may be abbreviated to any unique prefix (e.g. p for portrait).

The following options are recognized:

- -2 Print two logical pages per physical sheet ("two-up").
- -5152 Print file using emulation of PC graphics printer. Flags 2, c, f, i, I, L, O, R, w, and W are ignored.
- -cn Print n copies of each page of the document. Copies are not collated.
- e[where][,level]

Report errors to where. Where may be one of mail(default), message, or trailer. Level selects the severity threshold for error reports. Level may be one of none(default), error(errors only), warning(warnings and errors), or all(warnings, errors and informational messages). (Fatal errors and internal errors are always reported.)

-ffont Print the document in font font. The available fonts and code page tables are listed in "The IBM 3812 Pageprinter." Give the font name and size, and code page table if different from fcp. For example:

pprint -fPRESTIGE.9 filea

pprint -fCourier.B.10.acp filea

will print filea using the fonts PRESTIGE 9 (with the codepage fcp) and Courier Bold 10 (with the codepage acp) respectively.

- hheader

Use header as the page header for pr.

- -in Indent document by prepending n blanks to each line.
- -J Suppress printing of the job header page.
- $-\ln$ Adjust interline spacing to print *n* lines per page.
- Lorientation

Set page orientation (default portrait). Orientation may be one of portrait, inverted, left, right, or landscape; the default is right. Landscape orientations (left and right) default to a page width of 152 columns.

- mmessage

Print message on the job header page.

- n[filter]

Override the default input filter pr(1). The default for filter is cat(1).

$-\mathbf{O}[width]$

Outline pages. Optional width selects width of border in pels (default 3).

- Pprinter

Print this file on printer (default pp).

PPRINT(1) PPRINT(1)

- -parg Pass arg to the input filter. This may be a single parameter (e.g. -p-h) or a string (e.g. -p"-h-f-m").
- $-\mathbf{R}[n]$ Print page rules every n lines (default 2).
- -t[filter]

Override the default output filter lpr(1). The default for filter is cat(1).

- -v Send file containing Page Map Primitive (PMP) commands to output filter (*lpr*(1)). Flags 2, c, f, i, l, L, O, R, w, and W are ignored. PMP is described in the *IBM 3812 Pageprinter Programming Reference*.
- -wn Set the line width to n columns. A line width of more than 80 columns defaults to right landscape orientation.
- -W Wrap long lines. Multiple occurrences of -W toggle between wrap and truncate.

FILES

/usr/lib/font/dev3812/fonts/*.dat /usr/lib/font/dev3812/fonts/*.*cp /usr/adm/lpd-errs Font descriptions
Font codepage descriptions
Error message file (site dependent)

SEE ALSO

font3812(5), width3812(8)

IBM 3812 Pageprinter Programming Reference, S544-3268

"The IBM 3812 Pageprinter" in Volume II, Supplementary Documents

BUGS

The only setting of where that is implemented for the -e flag is mail.

PRFL(1)

NAME

prfl - IBM 4201 Proprinter/IBM 5152 Graphics Printer nroff post-processing filter

SYNOPSIS

prfl[-g]

DESCRIPTION

Prfl is a post-processing filter to convert the intermediate output of nroff(1) and colpro(1) to the IBM 4201 Proprinter control sequences and characters. In particular, the filter does the following:

- 1. Converts strings of the form _ < bs > < char > _ < bs > < char > to < start underlining > < char > < char > < stop underlining > .
- 2. Converts sequences of the form < Esc > Ax to IBM 4201 Proprinter nonprinting control commands.
- 3. Convert sequences of the form < Esc > [B,C]? to a set of sequences to download special characters. If the -g option is specified, these characters are printed in graphics mode.
- 4. Convert sequences of the form $\langle Esc \rangle Dx$ to x + 130 (nroff cannot send codes over 127).
- 5. Convert sequences of the form < Esc > Ex to print the code x using the all-characters font.
- 6. Convert sequences of the form $\langle Esc \rangle Fx$ to print the code x + 130 using the all-characters font.
- 7. Convert $\langle Esc \rangle 9$ to $\langle 1/2 \rangle$ space linefeed \rangle .

The -g option uses only those escape sequences understood by the IBM 5152 Graphics Printer. Prfl is provided primarily for use by proff(1).

SEE ALSO

colpro(1), nroff(1), proff(1), lpfilter(8R)

IBM 4201 Proprinter Guide to Operations, PN6328945

IBM Personal Computer Hardware Reference Library Guide to Operations, 1502490

PROFF(1) PROFF(1)

NAME

proff - nroff for the IBM 4201 Proprinter and IBM 5152 Graphics Printer

SYNOPSIS

proff
$$[-g][-t][-Pxx][-Lparms][nroff-options][files...]$$

DESCRIPTION

Proff formats text in the named files for the IBM 4201 Proprinter (default) and the IBM 5152 Graphics Printer. See also nroff(1).

If no *file* argument is present, the standard input is read. An argument consisting of a single minus (-) is taken to be a file name corresponding to standard input.

Options are:

- -g produce output for the IBM 5152 Graphics Printer
- -t send output to standard output.
- -Pxx send output to printer xx which corresponds to an entry in /etc/printcap. The default is taken from the PRINTER environment variable, if it exists; otherwise **lp** is used.
- Lparms

pass parameter parms to lpr(1). This may be a single parameter (e.g., -L-h) or a string (e.g., -L"-h -r -m").

All other options are passed to nroff.

FILES

/usr/lib/term/tabpro	IBM 4201 Proprinter driving tables
/usr/lib/term/tabgra	IBM 5152 Graphics Printer driving tables
/usr/lib/ibmlp/ibmpro	IBM 4201 Proprinter lpr filter
/usr/lib/ibmlp/ibmgra	IBM 5152 Graphics Printer lpr filter
/usr/adm/lpd-errs	Error message file (site dependent)

SEE ALSO

colpro(1), eqn(1), lpr(1), nroff(1), prfl(1), tbl(1), lpfilter(8R)

- J. F. Ossanna, "NROFF/TROFF User's Manual" in UNIX User's Supplementary Documents
- B. W. Kernighan, "A TROFF Tutorial" in UNIX User's Supplementary Documents

PTROFF(1) PTROFF(1)

NAME

ptroff - print troff files on IBM 3812 Pageprinter

SYNOPSIS

ptroff [options] [files]

DESCRIPTION

Ptroff prints troff source files on an IBM 3812 Pageprinter using |usr/ibm|troff. If no files are specified, ptroff reads the standard input.

Ptroff accepts a number of options which control job parameters, page layout, font selection, and input filtering; some flags are passed through to \(\lloss \frac{\lloss \lloss \lloss \lloss \lloss \lloss \rloss \rloss \rloss \lloss \rloss \rloss

The flags - [Faimnoqr] are passed to /usr/ibm/troff, and are documented elsewhere. The following flags are recognized by ptroff:

- -cn Print n copies of each page of the document. Copies are not collated.
- $-\mathbf{e}[where][,level]$

Report errors to where. Where may be one of mail(default), message, or trailer. Level selects the severity threshold for error reports. Level may be one of none(default), error(errors only), warning(warnings and errors), or all(warnings, errors, and informational messages).

- -E Run actual troff job "elsewhere". Usually run on the print server.
- hheader

Use *header* as the file name on the job header page.

- -J Suppress printing of the job header page.
- -L[orientation]

Set page orientation (default portrait). Orientation may be one of portrait, inverted, left, or right. The default for orientation is right. Left and right are landscape orientations.

-Mmessage

Print message on the job header page.

-**P**printer

Send output to printer.

-t Print the output of ditroff on the standard output. Do not send output to printer.

NOTES

The output of the -t command is an intermediate language produced by troff; it is not a text approximation of troff output.

BUGS

The only setting of where for the -e flag that is implemented is mail.

The $-\mathbf{E}$ flag is not fully implemented.

The -L flag is not fully implemented.

FILES

/usr/lib/font/dev3812/fonts/*.dat Font descriptions
/usr/lib/font/dev3812/fonts/*.*cp Font codepage descriptions

SEE ALSO

font3812(5), width3812(8),

"The IBM 3812 Pageprinter" in Volume II, Supplementary Documents

SCALE(1) SCALE(1)

NAME

scale - resize a bitmap image

SYNOPSIS

$$scale [-xmmm] [-ynnn] [-b] [-w]$$

DESCRIPTION

Scale reads a bitmap image file from the standard input, expands or contracts it to a specified size, and writes it to the standard output. The option -xmmm specifies an x-axis (horizontal) size of mmm pixels, and -ynnn specifies a y-axis (vertical) size of nnn scanlines. No scaling is done in a particular direction if the corresponding option is omitted.

The -b and -w options influence the x-axis compression algorithm to preserve black or white details respectively. They can be used singularly or together; which choice is appropriate is an esthetic judgment that depends on the nature of the image.

Scale is normally used by bitprt(1). A typical use would be

dumpaed | scale
$$-x700 - b - w$$
 | $1pr - Plp - h - v$

SEE ALSO

bitprt(1), dumpaed(1), dumpapa16(1), dumpapa8(1), dumpapa8c(1), lpr(1), lp(4), lpfilter(8R)

BUGS

Scale truncates the specified x-size to the nearest multiple of 8.

Options $-\mathbf{b}$ and $-\mathbf{w}$ have no effect in the vertical direction (y-axis).

SUPPORT(1) SUPPORT(1)

NAME

support - hardware and software support information

SYNOPSIS

support

DESCRIPTION

Support tells the user how to obtain information about IBM RT PC hardware and software support.

NAME

tar - tape archiver

SYNOPSIS

tar [key] [name ...]

DESCRIPTION

Tar saves and restores multiple files on a single file (usually a magnetic tape, but it can be any file). Tar's 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 tar are file or directory names specifying which files to dump or restore. In all cases, appearance of a directory name refers to the files and (recursively) subdirectories of that directory.

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

- r The named files are written on the end of the tape. The c function implies this.
- x The named files are extracted from the tape. If the named file matches a directory whose contents had been written onto the tape, this directory is (recursively) extracted. The owner, modification time, and mode are restored (if possible). If no file argument is given, the entire content of the tape is extracted. Note that if multiple entries specifying the same file are on the tape, the last one overwrites all earlier.
- t The names of the specified files are listed each time they occur on the tape. If no file argument is given, all of the names on the tape are listed.
- The named files are added to the tape if either they are not already there or have been modified since last put on the tape.
- c Create a new tape; writing begins on the beginning of the tape instead of after the last file. This command implies r.

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

- On output, tar normally places information specifying owner and modes of directories in the archive. Former versions of tar, when encountering this information will give error message of the form
 - "<name>/: cannot create".
 - This modifier will suppress the directory information.
- p This modifier says to restore files to their original modes, ignoring the present umask(2). Setuid and sticky information will also be restored to the super-user.
- 0, ..., 9 This modifier selects an alternate drive on which the tape is mounted. The default is drive 0 at 1600 bpi, which is normally /dev/rmt8.
- Normally tar does its work silently. The v (verbose) option makes tar print the name of each file it treats preceded by the function letter. With the t function, the verbose option gives more information about the tape entries than just their names.
- w Tar prints the action to be taken followed by file name, then wait for user confirmation. If a word beginning with 'y' is given, the action is done. Any other input means don't do it.
- f Tar uses the next argument as the name of the archive instead of /dev/rmt?. If the name of the file is '-', tar writes to standard output or reads from standard input, whichever is appropriate. Thus, tar can be used as the head or tail of a filter chain. Tar can also be used to move hierarchies with the command
 - cd fromdir; tar cf . | (cd todir; tar xf -)
- b Tar uses the next argument as the blocking factor for tape records. The default is 20 (the maximum). This option should only be used with raw magnetic tape archives (See f above). The block size is determined automatically when reading tapes (key letters

TAR(1) TAR(1)

'x' and 't').

tells tar to complain if it cannot resolve all of the links to the files dumped. If this is not specified, no error messages are printed.

- m tells tar not to restore the modification times. The modification time will be the time of extraction.
- h Force tar to follow symbolic links as if they were normal files or directories. Normally, tar does not follow symbolic links.
- B Forces input and output blocking to 20 blocks per record. This option was added so that tar can work across a communications channel where the blocking may not be maintained.
- C If a file name is preceded by -C, then tar will perform a chdir(2) to that file name. This allows multiple directories not related by a close common parent to be archived using short relative path names. For example, to archive files from /usr/include and from /etc, one might use

tar c -C /usr include -C / etc

Previous restrictions dealing with tar's inability to properly handle blocked archives have been lifted.

FILES

/dev/rmt? /tmp/tar*

SEE ALSO

tar(5)

DIAGNOSTICS

Complaints about bad key characters and tape read/write errors.

Complaints if enough memory is not available to hold the link tables.

BUGS

There is no way to ask for the *n*-th occurrence of a file.

Tape errors are handled ungracefully.

The u option can be slow.

The current limit on file name length is 100 characters.

There is no way selectively to follow symbolic links.

When extracting tapes created with the r or u options, directory modification times may not be set correctly.

TN3270(1)

NAME

tn3270 - full-screen remote login to IBM VM/CMS

SYNOPSIS

tn3270 [-d] [-n filename] [-t commandname] [sysname [port]]

DESCRIPTION

Tn3270 permits a full-screen, full-duplex connection from a machine running a UNIX operating system to an IBM (or compatible) machine. Tn3270 gives the appearance of being logged in to the remote machine from an IBM 3270 terminal. Of course, you must have an account on the machine to which you connect in order to log in. Tn3270 looks to the user in many respects like the Yale ASCII Terminal Communication System II. Tn3270 is actually a modification of the Arpanet TELNET user interface (see telnet(1)) which will, in certain circumstances, interpret and generate raw 3270 control streams.

The flags to tn3270 are as follows:

-d Turn on socket-level tracing (for super-user only)

-n filename

Specify a file to receive network trace data output (from commands "toggle netdata" and "toggle options", see *telnet*(1c)); the default is for output to be directed to the standard error file.

-t commandname

Specify a command to process IBM 4994 style transparent mode data received from the remote IBM machine.

sysname The name of the remote system. If the remote name is NOT specified, the user will be prompted for a command (see below).

port The port to connect to on the remote system. Normally, *tn3270* attempts to connect to the standard TELNET port (port 23) on the remote machine.

When tn3270 first connects to the remote system, it will negotiate to go into 3270 mode. Part of this negotiation involves telling the remote system what model 3270 it is emulating. In all cases, tn3270 emulates a 3278 terminal. To decide which specific model, tn3270 looks at the number of lines and columns on the actual terminal (as defined in the TERM environment variable; see termcap(5)). The terminal (or window in which tn3270 is running, on multiple window systems) must have at least 80 columns and 24 lines, or tn3270 will not go into emulation mode. If the terminal does have at least 80 columns and at least 24 lines, the following table describes the emulation:

minimum size (rows*columns)	emulated terminal
27*132	3278 model 5
43*80	3278 model 4
32*80	3278 model 3
24*80	3278 model 2.

Emulation of the 3270 terminal is done in the process. This emulation involves mapping 3270-style commands from the host into appropriate sequences to control the user's terminal screen. Tn3270 uses curses(3x) and the /etc/termcap file to do this. The emulation also involves simulating the special 3270 keyboard keys (program function keys, etc.) by mapping sequences of keystrokes from the ASCII keyboard into appropriate 3270 control strings. This mapping is terminal dependent and is specified in a description file, /etc/map3270, (see map3270(5)) or in an

TN3270(1) TN3270(1)

environment variable MAP3270 (and, if necessary, MAP3270A, MAP3270B, and so on - see mset(1)). Any special function keys on the ASCII keyboard are used whenever possible. If an entry for the user's terminal is not found, tn3270 looks for an entry for the terminal type unknown. If this is not found, tn3270 uses a default keyboard mapping (see map3270(5)).

The first character of each special keyboard mapping sequence is either an ASCII escape (ESC), a control character, or an ASCII delete (DEL). If the user types an unrecognized function key sequence, tn3270 sends an ASCII bell (BEL), or a visual bell if defined in the user's termcap entry, to the user's terminal and nothing is sent to the IBM host.

If tn3270 is invoked without specifying a remote host system name, it enters local command mode, indicated by the prompt "tn3270>". In this mode, tn3270 accepts and executes all the commands of telnet(1), plus one additional command, transcom, used to specify a command for IBM 4994-style transparent mode processing.

Tn3270 command mode may also be entered, after connecting to a host, by typing a special escape sequence. If tn3270 has succeeded in negotiating 3270 mode with the remote host, the escape sequence will be as defined by the map3270 (see map3270(5)) entry for the user's terminal type (typically control-C); otherwise the escape sequence will initially be set to the single character '^]' (control right square bracket).

While in command mode, any host login session is still alive but temporarily suspended. The host login session may be resumed by entering an empty line (press the RETURN key) in response to the command prompt. A session may be terminated by logging off the foreign host, or by typing "quit" or "close" while in local command mode.

FILES

/etc/termcap /etc/map3270

NOTES

The IBM 4994 style transparent mode command is invoked when tn3270 receives IBM 4994 style transparent output from the remote host. Output and input pipes are created for communication between the two processes. The pipes are closed when a 3270 clear command is received from the remote hosts, signalling the end of transparent mode output. Transparent mode is necessary for sending ASCII control characters over the 3270 terminal connection; ASCII graphics terminal support is accomplished this way. Developers of transcom commands should note that the transcom stdin pipe end will be in CBREAK mode, with ECHO and CRMOD turned off.

SEE ALSO

mset(1), telnet(1), curses(3x), termcap(3x), termcap(5), map3270(5), Yale ASCII Terminal Communication System II Program Description/Operator's Manual (IBM SB30-1911)

BUGS

Tn3270 is slow and uses system resources prodigiously.

Not all 3270 functions are supported, nor all Yale enhancements.

Error conditions (attempting to enter data in a protected field, for example) should cause a message to be sent to the user's terminal instead of just ringing a bell.

NAME

up, down - client Remote Virtual Disk (RVD) utilities

SYNOPSIS

```
up [ - adfrtxv ] [ -Rn ] [ -Tn ] [ name ... ] [ -p passwd ]
up [ -ftv ] [ -Rn ] [ -Tn ] -c pack driveno server mode mount-directory [passwd]
down [ - adrtxv ] [ name ... ]
```

DESCRIPTION

The up utility spins up and mounts RVD packs; the down utility unmounts and spins them down.

By default, *up* spins up and mounts packs based on information stored in the |etc/rvd/rvdtab| file. You can also use *up* to spin up and mount packs that are not specified in |etc/rvd/rvdtab|. See the -c option described below.

When up spins up a pack in exclusive mode, the utility automatically checks the pack's file system with the Unix file system checker fsck. up will skip this file system check if you use the $-\mathbf{f}$ option described below.

The options to up are as follows:

- -a up tries to spin up every pack specified in $\frac{|etc|rvd|rvdtab}{}$.
- -c Lets you specify an arbitrary rvdtab-style line to *up*. This is useful for spinning up and mounting a "one-time" pack that is not specified in /etc/rvd/rvdtab. Note that all fields must be specified, except the password field which you will want to specify only if you are using *up* in a shell script. If you are using *up* as a command to the shell, don't specify the password on the command line. *up* will prompt you for it after you enter the command.
- -d up tries to spin up every default pack specified in /etc/rvd/rvdtab. Default packs are marked by either a * or %.
- -f up skips the file system check on all selected packs spunup in exclusive mode (-x). Normally, all packs spunup in exclusive mode are checked and a failure results in the pack being not mounted and spundown.
- -t up processes the selected packs, then produces a listing of the approximate state of the client RVD system.

$-\mathbf{r}, -\mathbf{x}$

up spins up all selected packs in the given mode, overriding the default mode specified in |etc/rvd/rvdtab|. The |etc/rvd/rvdtab| entry must "allow" the given mode in order to be selected. Any number of these can be specified, but only the last one specified takes effect.

- -v Verbose mode. up displays a status message after processing each pack.
- -R# Retry count, where # specifies the number of times up will retry a pack that fails to spin up the first time. Those entries marked with a * will be retried if a failure occurred and some other pack was not spunup on the drive. If you use the -R option without providing a specific #, up will retry the pack an infinite number of times. If you provide a numerical argument, up will try to process the pack that number of times before giving up.
- -T# Time interval for retries, where # specifies the number of seconds between retry attempts. If you do not provide a specific #, up will retry the pack every 60 seconds.

The name arguments refer to pack names found in /etc/rvd/rvdtab. The pack name may be of the form "host:pack" where either "host:" or "pack" may be null. Up selects those entries in /etc/rvd/rvdtab that match as much of the pack name as specified.

If the pack you want to spin up requires a password, you will be prompted to supply it. For those of you who write shell scripts that use the up command, the -p option lets you specify pack passwords in command lines. Use it as follows:

up packname -p passwd

where packname specifies the rvd pack, and password specifies the pack's passwd.

The options to down are as follows:

- -a down tries to process every pack specified in /etc/rvd/rvdtab.
- -d down tries to process every default pack specified in /etc/rvd/rvdtab.
- -t down processes the selected packs, then produces a listing of the approximate state of the client RVD system.

$-\mathbf{r}, -\mathbf{x}$

Only one of these can be specified (as for up above).

The remaining arguments are pack names (as for up above).

DIAGNOSTICS

Intended to be self-explanatory.

Each server involved in an invocation of *up* is queried (by *rvdgetm*(8)) for an "RVD message of the day" (of sorts). These are displayed as each RVD server host is contacted to spinup a disk.

FILES

/etc/rvd/rvdtab

RVD client database

SEE ALSO

rvdtab(5), mtab(5), fsck(8), rvdgetm(8)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

VGRIND(1) VGRIND(1)

NAME

vgrind - grind nice listings of programs for the IBM 3812 Pageprinter

SYNOPSIS

```
vgrind [-f][-1][-t][-n][-x][-sn][-h header [-d file [-l anguage [-d] name ...
```

DESCRIPTION

(Note: Vgrind requires ptroff, a function of ditroff (device-independent troff), an optional, licensed feature of IBM Academic Information Systems 4.3.)

Vgrind formats the program sources which are arguments in a nice style using ptroff(1). Comments are placed in *italics*, keywords in **bold face**, and the name of the current function is listed down the margin of each page as it is encountered.

Vgrind runs in two basic modes, filter mode or regular mode. In filter mode, vgrind acts as a filter in a manner similar to tbl(1). The standard input is passed directly to the standard output except for lines bracketed by the troff-like macros:

- .vS starts processing
- .vE ends processing

These lines are formatted as described above. The output from this filter can be passed to ptroff for output. There need be no particular ordering with eqn(1) or tbl(1).

In regular mode, vgrind accepts input files, processes them, and passes them to ptroff(1) for output.

In both modes, vgrind passes any lines beginning with a decimal point without conversion.

The options are:

- f forces filter mode
- forces input to be taken from standard input (default if $-\mathbf{f}$ is specified)
- -t similar to the same option in ptroff causing formatted text to go to the standard output
- -n forces no keyword bolding
- -x outputs the index file in a "pretty" format. The index file itself is produced whenever vgrind is run with a file called index in the current directory. The index of function definitions can then be run off by giving vgrind the -x option and the file index as argument.
- -s specifies a point size to use on output (exactly the same as the argument of a .ps)
- -h specifies a particular header to put on every output page (default is the file name)
- -d specifies an alternate language definitions file (default is /usr/lib/vgrindefs)
- specifies the language to use. Currently known are PASCAL (-lp), MODEL (-lm), C (-lc or the default), CSH (-lcsh), SHELL (-lsh), RATFOR (-lr), and ICON (-ll).

FILES

index file where source for index is created

/usr/lib/tmac/tmac.vgrind macro package /usr/lib/vfontedpr preprocessor

/usr/lib/vgrindefs language descriptions

SEE ALSO

ptroff(1), vlp(1), vgrindefs(5)

BUGS

Vionted assumes a certain programming style is followed:

VGRIND(1) VGRIND(1)

For C – function names can be preceded on a line only by spaces, tabs, or an asterisk. The parenthesized arguments must also be on the same line.

For PASCAL – function names need to appear on the same line as the keywords function or procedure.

For MODEL – function names need to appear on the same line as the keywords is beginproc.

If these conventions are not followed, the indexing and marginal function name comment mechanisms will fail.

More generally, arbitrary formatting styles for programs do not look good. The use of spaces to align source code fails miserably; if you plan to *vgrind* your program, you should use tabs. This is somewhat inevitable since the font used by *vgrind* is variable width.

The mechanism of ctags in recognizing functions should be used here.

Filter mode does not work in documents using the -me or -ms macros.

VMSTAT(1) VMSTAT(1)

NAME

vmstat - report virtual memory statistics

SYNOPSIS

```
vmstat [ - fsi ] [ drives ] [ interval [ count ] ]
```

DESCRIPTION

Vmstat delves into the system and normally reports certain statistics kept about process, virtual memory, disk, trap and cpu activity. If given a -f argument, it instead reports on the number of forks and vforks since system startup and the number of pages of virtual memory involved in each kind of fork. If given a -s argument, it instead prints the contents of the sum structure, giving the total number of several kinds of paging related events which have occurred since boot. If given a -i argument, it instead reports on the number of interrupts taken by each device since system startup.

If none of these options are given, *vmstat* will report in the first line a summary of the virtual memory activity since the system has been booted. If *interval* is specified, then successive lines are summaries over the last *interval* seconds. "vmstat 5" will print what the system is doing every five seconds; this is a good choice of printing interval since this is how often some of the statistics are sampled in the system; others vary every second, running the output for a while will make it apparent which are recomputed every second. If a *count* is given, the statistics are repeated *count* times. The format fields are:

Procs: information about numbers of processes in various states.

```
r in run queue
```

b blocked for resources (i/o, paging, etc.)

w runnable or short sleeper (< 20 secs) but swapped

Memory: information about the usage of virtual and real memory. Virtual pages are considered active if they belong to processes which are running or have run in the last 20 seconds. A "page" here is pagesize(1) bytes.

```
avm active virtual pages fre size of the free list
```

Page: information about page faults and paging activity. These are averaged each five seconds, and given in units per second.

```
re page reclaims (simulating reference bits)
at pages attached (found in free list)
pi pages paged in
po pages paged out
fr pages freed per second
de anticipated short term memory shortfall
sr pages scanned by clock algorithm, per-second
```

hd: Disk operations per second (this field is system dependent). Typically paging is split across several of the available drives. The number under each of these is the unit number.

Faults: trap/interrupt rate averages per second over last 5 seconds.

```
in (non clock) device interrupts per second
sy system calls per second
```

cs cpu context switch rate (switches/sec)

Cpu: breakdown of percentage usage of CPU time

```
us user time for normal and low priority processes sy system time
```

VMSTAT(1) VMSTAT(1)

id

cpu idle

To force vmstat to display specific drives, their names may be supplied on the command line.

FILES

/dev/kmem, /vmunix

SEE ALSO

systat(1), iostat(1)

The sections starting with Section 7.9, "Monitoring System Performance" in "Installing and Operating 4.3/RT," in Volume II, Supplementary Documents.

NAME

Xibm - X Window System (version 11) Server for IBM equipment

SYNOPSIS

Xibm [options]

DESCRIPTION

X is a network-transparent windowing system developed at MIT which runs under several operating systems and environments. Xibm is an X server based on the MIT sample server described in xwindows(1).

A script interface for Xibm, which starts the server, provides a window manager, and opens some windows, is provided by xwindows(1).

Only options and functions specific to IBM X servers are described here. For complete information on the X server, see xwindows(1).

Xibm is a multi-screen server. One server can manipulate one or several screens simultaneously. Alternatively, you can invoke a different instantiation of the server for each screen (thereby rendering "screen" and "display" equivalent; remember that in X parlance, "display" means server and "screen" means physical device).

A multi-screen server will move the cursor from screen to screen by moving off the edge of one screen onto the edge of the logical next screen. Logical ordering of screens is controllable with command line options (see below).

OPTIONS

Options understood by X and documented in xwindows(1) are not described here. Options specific to Xibm are:

- -pckeys Reverses the interpretation of the CAPS-LOCK and CTRL keys, to emulate the key positions on a PC keyboard.
- -rtkeys Default. Interprets CAPS-LOCK and CTRL as defined on the IBM RT PC keyboard.
- -wrapx Allows the cursor to wrap from the extreme rightmost position of the rightmost screen to the leftmost position of the leftmost screen. Without this option, the cursor is blocked at the extreme left of the leftmost screen, and at the extreme right of the rightmost screen.
- -wrapy Allows the cursor to wrap from the top position of the highest screen to the bottom position of the lowest screen. Without this option, the cursor is blocked at the top of the top screen, and at the bottom of the bottom screen.
- -wrap Specifies both -wrapx and -wrapy.
- -8514 Instructs the server to use the IBM 8514/ Λ polychrome adapter and attached display, if installed. For use on a machine with an 8514/ Λ installed, this is the default.
- -vga Instructs the server to use the PS/2 planar Video Graphics Array polychrome display controller with its attached display, if installed.
- -mpel Instructs the server to use the IBM Megapel color adapter and attached 5081 display, if installed. For IBM RT PC machines with this device installed, this is the default.
- -ega Instructs the server to use the IBM 5154 and Enhanced Color Display, if installed.
- -apa16 Instructs the server to use the IBM 6155 Extended Monochrome Graphics Display, if installed.
- -aed Instructs the server to use the IBM Academic Information Systems experimental display, if installed. This display is not generally available.

ORDERING SCREENS

If the Xibm command is entered without requesting a particular screen, the first screen found from the following list will be used:

XIBM(1)

8514 vga mpel ega apa16

If screens are specified, then the screens will be logically ordered, from left to right, in the order they are specified on the command line. For example:

Xibm -apa16 -mpel

will start Xibm on two screens with :0.0 on the 6155 in the logical left position, and :0.1 on the 5081 in the logical right position. The cursor will hop from the right edge of the 6155 to the left edge of the 5081, and vice versa. For explanation of the terminology ":X.Y", see xwindows(1).

SCREEN CHARACTERISTICS

The 8514 is 1024x768 pixels, 256 or 16 displayable colors (depending on installed memory option) from a palette of 265K possible colors. With a greyscale display attached, only 64 distinct shades of grey can be displayed. 15-inch color (8514) or 16-inch greyscale (8604) displays are supported.

The VGA is 640x480 pixels, with 16 displayable colors or greys from a palette of 256K possible colors or 64 possible greys. 13-inch color (8513) or greyscale (8503) are supported.

The 5081 is 1024x1024 pixels with 256 displayable colors from a palette of 4096 possible colors. The 5081 is supported in both 16- and 19-inch models.

The 5154 is 640x480 pixels, with 16 colors, from a palette of 64 possible colors.

The 6155 is 1024x768 pixels, monochrome.

The IBM Academic Information Systems experimental display is 1024x800, monochrome.

SEE ALSO

```
uwm(1), xinit(1), xwindows(1), intro(3X), X(8C)
The X Window System (Scheifler, Gettys, July 1986)
```

BUGS

Moving windows from screen to screen is not supported.

Starting two servers on the same screen can lead to interesting and unpredictable results.

XWINDOWS(1) XWINDOWS(1)

NAME

xwindows - start Xibm, uwm window manager, two xterms, and xclock

SYNOPSIS

xwindows [:number] [screen ...] [-w window-manager] [-fn fontname] [-pckcys]

DESCRIPTION

Xwindows is a shell script that starts up the Xibm server, uwm, two xterms, and an xclock on the specified screen. Before using xwindows, make sure you have copied /usr/lib/X11/uwm/.uwmrc into your home directory; uwm will it need for its menus, etc.

Number is an integer in the range 0-7 inclusive indicating the display number; for example, :0. If no number is given the shell variable DISPLAY is used. A number given on the command line, however, will override a DISPLAY variable defined in the shell environment.

The following screens are recognized:

- -8514 The IBM 8514 PS/2 Color Display (IBM 8514/A Adapter required)
- -vga The IBM Video Graphics Array (VGA) Display
- -mpel The IBM 5081 Display with MegaPel adapter
- -ega The IBM 5154 Enhanced Color Display with adapter
- -apa16 The IBM 6155 Extended Monochrome Graphics Display with adapter
- -aed The IBM Academic Information Systems Experimental Display

If no screens are specified the server will start on the first available screen chosen from the above list. The search order is from top to bottom of the list.

The following options are each recognized as a separate argument:

- -w Xwindows uses the next argument as the window manager instead of uwm.
- -fn Xwindows uses the next argument as the name of the font instead of the default font.
- -pckeys This modifier causes the < Caps Lock > and < Ctrl > keys to be switched while xwindows is being run to make their positions more familiar to touch typists.

EXAMPLES

xwindows

xwindows :0 -8514

xwindows:1-vga-fn fixed-pckeys

FILES

/ibm/acis/usr/guest/guest/xwindows /ibm/acis/usr/guest/guest/src/*.c /ibm/acis/usr/guest/guest/lib/* /ibm/acis/usr/guest/guest/.uwmrc /ibm/acis/usr/guest/guest/help/* xwindows executable file source for executables used by uwm menus executables used by uwm menus read by uwm when it starts up help scripts used by uwm

SEE ALSO

andrew(1), uwm(1), Xibm(1), xinit(1)

"The X Windowing System" in IBM Academic Operating System 4.3, Volume II

Section 2. System Calls

This section describes system calls. Man pages found in IBM/4.3, but not in 4.3BSD, are marked with an asterisk (*).

- intro
- getfloatstate*
- getfpemulator*
- ptrace
- sigvec
- vdspin*
- vdstats*

IBM/4.3 Manual Pages

NAME

intro – introduction to system calls and error numbers

SYNOPSIS

#include < sys/errno.h >

DESCRIPTION

This section describes all of the system calls. Most of these calls have one or more error returns. An error condition is indicated by an otherwise impossible return value. This is almost always — 1; the individual descriptions specify the details.

As with normal arguments, all return codes and values from functions are of type integer unless otherwise noted. An error number is also made available in the external variable *errno*, which is not cleared on successful calls. Thus *errno* should be tested only after an error has occurred.

The following is a complete list of the errors and their names as given in $\langle sys/errno.h \rangle$.

0 Error 0 Unused.

1 EPERM Not owner

Typically this error indicates an attempt to modify a file in some way forbidden except to its owner or super-user. It is also returned for attempts by ordinary users to do things allowed only to the super-user.

2 ENOENT No such file or directory

This error occurs when a file name is specified and the file should exist but doesn't, or when one of the directories in a path name does not exist.

3 ESRCH No such process

The process whose number was given to kill and ptrace does not exist, or is already dead.

4 EINTR Interrupted system call

An asynchronous signal (such as interrupt or quit), which the user has elected to catch, occurred during a system call. If execution is resumed after processing the signal, it will appear as if the interrupted system call returned this error condition.

5 EIO I/O error

Some physical I/O error occurred during a read or write. This error may in some cases occur on a call following the one to which it actually applies.

6 ENXIO No such device or address

I/O on a special file refers to a subdevice which does not exist, or beyond the limits of the device. It may also occur when, for example, an illegal tape drive unit number is selected or a disk pack is not loaded on a drive.

7 E2BIG Arg list too long

An argument list longer than 20480 bytes is presented to execve.

8 ENOEXEC Exec format error

A request is made to execute a file which, although it has the appropriate permissions, does not start with a valid magic number, see a.out(5).

9 EBADF Bad file number

Either a file descriptor refers to no open file, or a read (resp. write) request is made to a file which is open only for writing (resp. reading).

10 ECHILD No children

Wait and the process has no living or unwaited-for children.

11 EAGAIN No more processes

In a fork, the system's process table is full or the user is not allowed to create any more processes.

12 ENOMEM Not enough core

During an execve or break, a program asks for more core or swap space than the system is able to supply. A lack of swap space is normally a temporary condition, however a lack of core is not a temporary condition; the maximum size of the text, data, and stack segments is a system parameter.

13 EACCES Permission denied

An attempt was made to access a file in a way forbidden by the protection system.

14 EFAULT Bad address

The system encountered a hardware fault in attempting to access the arguments of a system call.

15 ENOTBLK Block device required

A plain file was mentioned where a block device was required, e.g. in mount.

16 EBUSY Mount device busy

An attempt to mount a device that was already mounted or an attempt was made to dismount a device on which there is an active file directory. (open file, current directory, mounted-on file, active text segment).

17 EEXIST File exists

An existing file was mentioned in an inappropriate context, e.g. link.

18 EXDEV Cross-device link

A hard link to a file on another device was attempted.

19 ENODEV No such device

An attempt was made to apply an inappropriate system call to a device; e.g. read a write-only device.

20 ENOTDIR Not a directory

A non-directory was specified where a directory is required, for example in a path name or as an argument to *chdir*.

21 EISDIR Is a directory

An attempt to write on a directory.

22 EINVAL Invalid argument

Some invalid argument: dismounting a non-mounted device, mentioning an unknown signal in *signal*, reading or writing a file for which *seek* has generated a negative pointer. Also set by math functions, see *intro*(3).

23 ENFILE File table overflow

The system's table of open files is full, and temporarily no more opens can be accepted.

24 EMFILE Too many open files

Customary configuration limit is 20 per process.

25 ENOTTY Not a typewriter

The file mentioned in an *loctl* is not a terminal or one of the other devices to which these calls apply.

26 ETXTBSY Text file busy

An attempt to execute a pure-procedure program which is currently open for writing (or reading!). Also an attempt to open for writing a pure-procedure program that is being executed.

27 EFBIG File too large

The size of a file exceeded the maximum (about 10⁹ bytes).

28 ENOSPC No space left on device

During a write to an ordinary file, there is no free space left on the device.

29 ESPIPE Illegal seek

An Iseek was issued to a pipe. This error may also be issued for other non-seekable devices.

30 EROFS Read-only file system

An attempt to modify a file or directory was made on a device mounted read-only.

31 EMLINK Too many links

An attempt to make more than 32767 hard links to a file.

32 EPIPE Broken pipe

A write on a pipe or socket for which there is no process to read the data. This condition normally generates a signal; the error is returned if the signal is ignored.

33 EDOM Math argument

The argument of a function in the math package (3M) is out of the domain of the function.

34 ERANGE Result too large

The value of a function in the math package (3M) is unrepresentable within machine precision.

35 EWOULDBLOCK Operation would block

An operation which would cause a process to block was attempted on a object in non-blocking mode (see *ioctl* (2)).

36 EINPROGRESS Operation now in progress

An operation which takes a long time to complete (such as a *connect* (2)) was attempted on a non-blocking object (see *ioctl* (2)).

37 EALREADY Operation already in progress

An operation was attempted on a non-blocking object which already had an operation in progress.

38 ENOTSOCK Socket operation on non-socket

Self-explanatory.

39 EDESTADDRREQ Destination address required

A required address was omitted from an operation on a socket.

40 EMSGSIZE Message too long

A message sent on a socket was larger than the internal message buffer.

41 EPROTOTYPE Protocol wrong type for socket

A protocol was specified which does not support the semantics of the socket type requested. For example you cannot use the ARPA Internet UDP protocol with type SOCK STREAM.

42 ENOPROTOOPT Bad protocol option

A bad option was specified in a getsockopt(2) or setsockopt(2) call.

43 EPROTONOSUPPORT Protocol not supported

The protocol has not been configured into the system or no implementation for it exists.

44 ESOCKTNOSUPPORT Socket type not supported

The support for the socket type has not been configured into the system or no implementation for it exists.

45 EOPNOTSUPP Operation not supported on socket

For example, trying to open a socket.

46 EPFNOSUPPORT Protocol family not supported

The protocol family has not been configured into the system or no implementation for it exists.

47 EAFNOSUPPORT Address family not supported by protocol family

An address incompatible with the requested protocol was used. For example, you shouldn't necessarily expect to be able to use PUP Internet addresses with ARPA Internet protocols.

48 EADDRINUSE Address already in use

Only one usage of each address is normally permitted.

49 EADDRNOTAVAIL Can't assign requested address

Normally results from an attempt to create a socket with an address not on this machine.

50 ENETDOWN Network is down

A socket operation encountered a dead network.

51 ENETUNREACH Network is unreachable

A socket operation was attempted to an unreachable network.

52 ENETRESET Network dropped connection on reset

The host you were connected to crashed and rebooted.

53 ECONNABORTED Software caused connection abort

A connection abort was caused internal to your host machine.

54 ECONNRESET Connection reset by peer

A connection was forcibly closed by a peer. This normally results from the peer executing a shutdown (2) call.

55 ENOBUFS No buffer space available

An operation on a socket or pipe was not performed because the system lacked sufficient buffer space.

56 EISCONN Socket is already connected

A connect request was made on an already connected socket; or, a sendto or sendmsg request on a connected socket specified a destination other than the connected party.

57 ENOTCONN Socket is not connected

An request to send or receive data was disallowed because the socket is not connected.

58 ESHUTDOWN Can't send after socket shutdown

A request to send data was disallowed because the socket had already been shut down with a previous shutdown(2) call.

- 59 unused
- 60 ETIMEDOUT Connection timed out

A connect request failed because the connected party did not properly respond after a period of time. (The timeout period is dependent on the communication protocol.)

61 ECONNREFUSED Connection refused

No connection could be made because the target machine actively refused it. This usually results from trying to connect to a service which is inactive on the foreign host.

62 ELOOP Too many levels of symbolic links

A path name lookup involved more than 8 symbolic links.

63 ENAMETOOLONG File name too long

A component of a path name exceeded 255 characters, or an entire path name exceeded 1023 characters.

64 ENOTEMPTY Directory not empty

A directory with entries other than "." and ".." was supplied to a remove directory or rename call.

70 EVDBAD RVD-related disk error

The drive does not exist, the access mode is bad, or the drive is already spun up.

DEFINITIONS

Process ID

Each active process in the system is uniquely identified by a positive integer called a process ID. The range of this ID is from 0 to {PROC MAX}.

Parent process ID

A new process is created by a currently active process; see fork(2). The parent process ID of a process is the process ID of its creator.

Process Group ID

Each active process is a member of a process group that is identified by a positive integer called the process group ID. This is the process ID of the group leader. This grouping permits the signalling of related processes (see killpg(2)) and the job control mechanisms of csh(1).

Tty Group ID

Each active process can be a member of a terminal group that is identified by a positive integer called the tty group ID. This grouping is used to arbitrate between multiple jobs contending for the same terminal; see csh(1), and tty(4).

Real User ID and Real Group ID

Each user on the system is identified by a positive integer termed the real user ID.

Each user is also a member of one or more groups. One of these groups is distinguished from others and used in implementing accounting facilities. The positive integer corresponding to this distinguished group is termed the real group ID.

All processes have a real user ID and real group ID. These are initialized from the equivalent attributes of the process which created it.

Effective User Id, Effective Group Id, and Access Groups

Access to system resources is governed by three values: the effective user ID, the effective group ID, and the group access list.

The effective user ID and effective group ID are initially the process's real user ID and real group ID respectively. Either may be modified through execution of a set-user-ID or set-group-ID file (possibly by one its ancestors); see execve(2).

The group access list is an additional set of group ID's used only in determining resource accessibility. Access checks are performed as described below in "File Access Permissions".

Super-user

A process is recognized as a *super-user* process and is granted special privileges if its effective user ID is 0.

Special Processes

The processes with a process ID's of 0, 1, and 2 are special. Process 0 is the scheduler. Process 1 is the initialization process *init*, and is the ancestor of every other process in the system. It is used to control the process structure. Process 2 is the paging daemon.

Descriptor

An integer assigned by the system when a file is referenced by open(2), dup(2), or pipe(2) or a socket is referenced by socket(2) or socketpair(2) which uniquely identifies an access path to that file or socket from a given process or any of its children.

File Name

Names consisting of up to {FILENAME_MAX} characters may be used to name an ordinary file, special file, or directory.

These characters may be selected from the set of all ASCII character excluding 0 (null) and the ASCII code for / (slash). (The parity bit, bit 8, must be 0.)

Note that it is generally unwise to use *, ?, [or] as part of file names because of the special meaning attached to these characters by the shell.

Path Name

A path name is a null-terminated character string starting with an optional slash (/), followed by zero or more directory names separated by slashes, optionally followed by a file name. The total length of a path name must be less than {PATHNAME_MAX} characters.

If a path name begins with a slash, the path search begins at the *root* directory. Otherwise, the search begins from the current working directory. A slash by itself names the root directory. A null pathname refers to the current directory.

Directory

A directory is a special type of file which contains entries which are references to other files. Directory entries are called links. By convention, a directory contains at least two links, and .., referred to as *dot* and *dot-dot* respectively. Dot refers to the directory itself and dot-dot refers to its parent directory.

Root Directory and Current Working Directory

Each process has associated with it a concept of a root directory and a current working directory for the purpose of resolving path name searches. A process's root directory need not be the root directory of the root file system.

File Access Permissions

Every file in the file system has a set of access permissions. These permissions are used in determining whether a process may perform a requested operation on the file (such as opening a file for writing). Access permissions are established at the time a file is created. They may be changed at some later time through the *chmod*(2) call.

File access is broken down according to whether a file may be: read, written, or executed. Directory files use the execute permission to control if the directory may be searched.

File access permissions are interpreted by the system as they apply to three different classes of users: the owner of the file, those users in the file's group, anyone else. Every file has an independent set of access permissions for each of these classes. When an access check is made, the system decides if permission should be granted by checking the access information applicable to the caller.

Read, write, and execute/search permissions on a file are granted to a process if:

The process's effective user ID is that of the super-user.

The process's effective user ID matches the user ID of the owner of the file and the owner permissions allow the access.

The process's effective user ID does not match the user ID of the owner of the file, and either the process's effective group ID matches the group ID of the file, or the group ID of the file is in the process's group access list, and the group permissions allow the access.

Neither the effective user ID nor effective group ID and group access list of the process match the corresponding user ID and group ID of the file, but the permissions for "other users" allow access.

Otherwise, permission is denied.

Sockets and Address Families

A socket is an endpoint for communication between processes. Each socket has queues for sending and receiving data.

Sockets are typed according to their communications properties. These properties include whether messages sent and received at a socket require the name of the partner, whether communication is reliable, the format used in naming message recipients, etc.

Each instance of the system supports some collection of socket types; consult socket(2) for more information about the types available and their properties.

Each instance of the system supports some number of sets of communications protocols. Each protocol set supports addresses of a certain format. An Address Family is the set of addresses for a specific group of protocols. Each socket has an address chosen from the address family in which the socket was created.

SEE ALSO

intro(3), perror(3)

This page intentionally left blank.

```
NAME
        getfloatstate - return machine and process floating point state
SYNOPSIS
        #include < machine/float.h >
        int getfloatstate(state, size)
                                                   /* structure to hold return information */
        struct floatstate *state;
                                  *size:
                                                   /* Length */
        int setfloatstate(mask, state, size)
        struct floatstate *mask,
                                                   /* mask of bits/fields to change */
                                                   /* new fields/return state */
                                  *state;
                                                   /* Length */
                                  *size;
        int
        int whichfpa();
```

DESCRIPTION

Getfloatstate can be used to determine what floating point hardware exists on the machine, as well as what floating point hardware the current (invoking) process is using.

On input, size contains the length of the floatstate structure. On output, size contains the length of the returned structure.

Setfloatstate is used to change the floating point state. A user process may use setfloatstate to release any resources allocated to that process. In addition, the superuser may use setfloatstate to change the effective configuration of the actual hardware. This latter use is most useful when loading the Advanced Floating Point Adapter card. Setfloatstate returns, after processing, the new state in the floatstate structure pointed to by the state argument. In addition, the size of the returned floatstate structure is returned in the integer pointed to by the size argument.

Whichfpa returns a value indicating what floating point hardware the current (invoking) process is using. The returned values are FLOAT_MC881, FLOAT_FPA, FLOAT_AFPA, and FLOAT_EMUL.

The include file < machine/float.h > contains the definition of the floatstate structure, as well as definitions of the bits used to specify the hardware. Currently, the definitions are as follows:

```
struct floatstate {
       int
               hardware state,
                                      /* What exists on the system */
                                      /* What is allocated to this process */
               process state;
       void
               (*emulator)(),
                                      /* Address of floating point emulator */
                                     /* Address of floating point code generator */
               (*code generator)();
       int
               fpa_registerset; /* (At this time) which fpa register set */
#define FLOAT MC881
                                             /* Using mc881/mc881 exists */
                                      0x01
#define FLOAT_FPA
                                              /* Using fpa/afpa exists */
                                      0x02
#define FLOAT AFPA
                                             /* Using afpa/afpa operational*/
                                      0x04
#define FLOAT EMUL
                                      0x08
                                             /* Using emulator */
#define FLOAT_AFPA_HARDWARE 0x10
                                             /* Afpa hardware exists */
        * Control store accesses to afpa enabled.
        * In this state, the only legal afpa commands are
        * those to load the microcode.
#define FLOAT AFPA CONTROL STORE ENABLE 0x20
```

NOTES

The following macros, which are defined in < machine/float.h>, are for querying the floating point state information. Each of these takes an integer parameter.

```
float_has_881 (x)
float_has_fpa (x)
float_has_afpa (x)
float_has_fpa_or_afpa (x)
float_has_emul (x)
```

RETURN VALUE

If the calls succeed, a value of 0 is returned. If an error occurs, the value -1 is returned, and a more precise error code is placed in the global variable *errno*.

ERRORS

The possible errors are:

[EFAULT] The state parameter or size parameter specified a bad address.

[EINVAL] The size parameter pointed to a location which contained an unknown value.

SEE ALSO

getfpemulator(2)

"Floating Point Arithmetic" in Volume II, Supplementary Documents

BUGS

Setfloatstate disallows some changes in the floating point state silently, with no changes to errno. The only way to be sure that the new floating point state is what was desired is to interrogate the state returned in the floatstate structure pointed to by the state argument.

Setfloatstate allows the superuser to acquire a specific AFPA register set, even if the floating point state of the system is such that the system would not normally allow use of the floating point hardware. If the process loses the register set (for example, by a debugger looking at the contents of the registers), accesses to the register may be lost.

NAME

getfpemulator - return address of the floating-point emulator

SYNOPSIS

```
#include < syscall.h >
#include < sys/types.h >
(caddr_t) syscall (SYS_getfpemulator)
```

DESCRIPTION

Getspermulator can be used to determine the address of the floating-point emulator in the kernel, and whether a Floating Point Accelerator (FPA) card is present. The result is a function pointer to the floating-point emulator if no FPA is present; otherwise, a pointer to 0.

SEE ALSO

"IBM/4.3 Linkage Convention" in Volume II, Supplementary Documents

This page intentionally left blank.

PTRACE(2) PTRACE(2)

NAME

ptrace - process trace

SYNOPSIS

#include < signal.h >
#include < sys/ptrace.h >
ptrace(request, pid, addr, data)
int request, pid, *addr, data;

DESCRIPTION

Ptrace provides a means by which a parent process may control the execution of a child process, and examine and change its core image. Its primary use is for the implementation of breakpoint debugging. There are four arguments whose interpretation depends on a request argument. Generally, pid is the process ID of the traced process, which must be a child (no more distant descendant) of the tracing process. A process being traced behaves normally until it encounters some signal whether internally generated like "illegal instruction" or externally generated like "interrupt". See sigvec(2) for the list. Then the traced process enters a stopped state and its parent is notified via wait(2). When the child is in the stopped state, its core image can be examined and modified using ptrace. If desired, another ptrace request can then cause the child either to terminate or to continue, possibly ignoring the signal.

The value of the request argument determines the precise action of the call:

PT_TRACE_ME

This request is the only one used by the child process; it declares that the process is to be traced by its parent. All the other arguments are ignored. Peculiar results will ensue if the parent does not expect to trace the child.

PT READ I,PT READ D

The 32-bit word in the child process's address space at addr is returned. If I and D space are separated (e.g. historically on a pdp-11), request PT_READ_I indicates I space, PT_READ_D D space. Addr must be word-aligned. The child must be stopped. The input data is ignored.

PT READ U

The word of the system's per-process data area corresponding to addr is returned. Addr must be word-aligned and less than NBPG*UPAGES (currently 6114). This space contains the registers and other information about the process; its layout corresponds to kernel stack, followed by the *user* structure in the system. Note that on the IBM RT PC the stack is below the *user* structure rather than above it (in order to better detect a kernel stack overflow). The process's registers can be located by first reading u_ar0 from the *user* structure and performing the appropriate calculation.

PT WRITE I,PT WRITE D

The given data is written at the word in the process's address space corresponding to addr, which must be word-aligned. No useful value is returned. If I and D space are separated, request PT_WRITE_I indicates I space, PT_WRITE_D D space. Attempts to write in pure procedure fail if another process is executing the same file.

PT WRITE U

The process's system data is written, as it is read with request PT_READ_U. Only a few locations can be written in this way: the general registers, the multiply quotient register, and certain bits of the Interrupt Control and Status register.

PT CONTINUE

The data argument is taken as a signal number and the child's execution continues at location addr as if it had incurred that signal. Normally the signal number will be either 0 to indicate that the signal that caused the stop should be ignored, or that value fetched out of

PTRACE(2) PTRACE(2)

the process's image indicating which signal caused the stop. If addr is (int *)1 then execution continues from where it stopped.

PT KILL

The traced process terminates.

PT_STEP

Execution continues as in request PT_CONTINUE; however, as soon as possible after execution of at least one instruction, execution stops again. The signal number from the stop is SIGTRAP. (On the IBM RT PC, the level 0 interrupt request is used and just one instruction is executed.) This is part of the mechanism for implementing breakpoints.

PT READ F

The addr argument is taken as a floating point register number and the contents of that register are returned. The register number must be in the range supported by the actual floating point hardware (or emulator) in use by the traced process.

PT_WRITE_F

The given data are written into the floating point register corresponding to addr (0...15).

As indicated, these calls (except for request PT_TRACE_ME) can be used only when the subject process has stopped. The wait call is used to determine when a process stops; in such a case the "termination" status returned by wait has the value 0177 to indicate stoppage rather than genuine termination.

To forestall possible fraud, ptrace inhibits the set-user-id and set-group-id facilities on subsequent execve(2) calls. If a traced process calls execve, it will stop before executing the first instruction of the new image showing signal SIGTRAP.

RETURN VALUE

The call succeeds if the returned value is not -1 and the global variable *errno* is 0. If the call fails then a - 1 is returned and *errno* is set to indicate the error.

ERRORS

[EINVAL]	The request code is invalid.
IEINVALI	The specified process does no

[EINVAL] The specified process does not exist.

[EINVAL] The given signal number is invalid.

[EINVAL] The given signal number is invalid.

[EFAULT] The specified address is out of bounds.

[EPERM] The specified process cannot be traced.

SEE ALSO

adb(1), sigvec(2), wait(2)

BUGS

Ptrace is unique and arcane; it should be replaced with a special file which can be opened and read and written. The control functions could then be implemented with *ioctl(2)* calls on this file. This would be simpler to understand and have much higher performance.

The request 0 call should be able to specify signals which are to be treated normally and not cause a stop. In this way, for example, programs with simulated floating point (which use "illegal instruction" signals at a very high rate) could be efficiently debugged.

The error indication, -1, is a legitimate function value; errno, see intro(2), can be used to disambiguate.

It should be possible to stop a process on occurrence of a system call; in this way a completely controlled environment could be provided.

SIGVEC(2) SIGVEC(2)

DESCRIPTION

The system defines a set of signals that may be delivered to a process. Signal delivery resembles the occurrence of a hardware interrupt: the signal is blocked from further occurrence; the current process context is saved, and a new one is built. A process may specify a handler to which a signal is delivered, or specify that a signal is to be blocked or ignored. A process may also specify that a default action is to be taken by the system when a signal occurs. Normally, signal handlers execute on the current stack of the process. This may be changed, on a per-handler basis, so that signals are taken on a special signal stack.

All signals have the same *priority*. Signal routines execute with the signal that caused their invocation *blocked*, but other signals may yet occur. A global *signal mask* defines the set of signals currently blocked from delivery to a process. The signal mask for a process is initialized from that of its parent (normally 0). It may be changed with a *sigblock*(2) or *sigsetmask*(2) call, or when a signal is delivered to the process.

When a signal condition arises for a process, the signal is added to a set of signals pending for the process. If the signal is not currently blocked by the process, it is delivered to the process. When a signal is delivered, the current state of the process is saved, a new signal mask is calculated (as described below), and the signal handler is invoked. The call to the handler is arranged so that if the signal handling routine returns normally the process will resume execution in the context from before the signal's delivery. If the process wishes to resume in a different context, then it must arrange to restore the previous context itself.

When a signal is delivered to a process a new signal mask is installed for the duration of the process' signal handler (or until a *sigblock* or *sigsetmask* call is made). This mask is formed by taking the current signal mask, adding the signal to be delivered, and *or*'ing in the signal mask associated with the handler to be invoked.

Sigvec assigns a handler for a specific signal. If vec is non-zero, it specifies a handler routine and mask to be used when delivering the specified signal. Further, if the SV_ONSTACK bit is set in sv_flags, the system will deliver the signal to the process on a signal stack, specified with sig-stack(2). If ovec is non-zero, the previous handling information for the signal is returned to the user.

The following is a list of all signals with names as defined in the include file < signal.h>:

```
SIGHUP
             1
                  hangup
             2
SIGINT
                  interrupt
SIGOUIT
             3*
                  quit
SIGILL
             4*
                 illegal instruction
             5*
SIGTRAP
                 trace trap
SIGIOT
             6*
                  IOT instruction
                 EMT instruction
SIGEMT
```

SIGVEC(2) SIGVEC(2)

```
SIGFPE
                  floating point exception
SIGKILL
                  kill (cannot be caught, blocked, or ignored)
              10* bus error
SIGBUS
              11* segmentation violation
SIGSEGV
SIGSYS
              12* bad argument to system call
                  write on a pipe with no one to read it
SIGPIPE
              13
SIGALRM
              14
                  alarm clock
SIGTERM
              15
                  software termination signal
SIGURG
              16• urgent condition present on socket
              17† stop (cannot be caught, blocked, or ignored)
SIGSTOP
SIGTSTP
              18† stop signal generated from keyboard
              19• continue after stop (cannot be blocked)
SIGCONT
SIGCHLD
              20 child status has changed
SIGTTIN
              21† background read attempted from control terminal
              22† background write attempted to control terminal
SIGTTOU
SIGIO
              23• I/O is possible on a descriptor (see fcntl(2))
SIGXCPU
              24
                  cpu time limit exceeded (see setrlimit(2))
SIGXFSZ
              25
                  file size limit exceeded (see setrlimit(2))
                  virtual time alarm (see setitimer(2))
SIGVTALRM 26
                  profiling timer alarm (see setitimer(2))
SIGPROF
              27
SIGWINCH
              28• window size change
SIGUSR1
                  user defined signal 1
              30
SIGUSR2
                  user defined signal 2
```

The starred (*) signals in the list above cause a core dump if not caught or ignored.

Once a signal handler is installed, it remains installed until another sigvec call is made, or an execve(2) is performed. The default action for a signal may be reinstated by setting sv_handler to SIG_DFL; this default is termination (with a core image for starred signals) except for signals marked with • or †. Signals marked with • are discarded if the action is SIG_DFL; signals marked with † cause the process to stop. If sv_handler is SIG_IGN the signal is subsequently ignored, and pending instances of the signal are discarded.

If a caught signal occurs during certain system calls, the call is normally restarted. The call can be forced to terminate prematurely with an EINTR error return by setting the SV_INTERRUPT bit in sv_flags. The affected system calls are read(2) or write(2) on a slow device (such as a terminal; but not a file) and during a wait(2).

After a fork(2) or vfork(2) the child inherits all signals, the signal mask, the signal stack, and the restart/interrupt flags.

Execve(2) resets all caught signals to default action and resets all signals to be caught on the user stack. Ignored signals remain ignored; the signal mask remains the same; signals that interrupt system calls continue to do so.

NOTES

The mask specified in vec is not allowed to block SIGKILL, SIGSTOP, or SIGCONT. This is done silently by the system.

The SV_INTERRUPT flag is not available in 4.2BSD, hence it should not be used if backward compatibility is needed.

RETURN VALUE

A zero value means that the call succeeded. A -1 return value means an error occurred and error is set to show the reason.

ERRORS

Sigvec fails with no new signal handler installed when one of the following occurs:

SIGVEC(2) SIGVEC(2)

[EFAULT] Either vec or ovec points to memory that is not a valid part of the process address space.
 [EINVAL] Sig is not a valid signal number.
 [EINVAL] An attempt is made to ignore or supply a handler for SIGKILL or SIGSTOP.
 [EINVAL] An attempt is made to ignore SIGCONT (by default SIGCONT is ignored).

SEE ALSO

kill(1), ptrace(2), kill(2), sigblock(2), sigsetmask(2), sigpause(2), sigstack(2), sigvec(2), setjmp(3), siginterrupt(3), tty(4)

NOTES (IBM RT PC)

The handler routine can be declared:

handler(sig, code, scp) int sig, code; struct sigcontext *scp;

Here sig is the signal number, into which the hardware faults and traps are mapped as defined below. Code is a parameter that further defines the signal on SIGILL or SIGFPE; it is zero in all other cases. For SIGILL, code is the value of the hardware machine/program check register (mcs_pcs) at the time of the trap; the code for SIGFPE is as given below. Scp is a pointer to the sigcontext structure (defined in < signal.h >), used to restore the context from before the signal. It contains all the context information needed to resume execution at the point that the signal was delivered.

This context includes the general registers (in sc_regs[0] through sc_regs[15]), the Instruction Adresss Register (IAR) in sc_iar, the Interrupt Status and Condition Code (ICSCS) in sc_icscs, the Multiply Quotient register (MQ) (in sc_regs[17]), and if the SC_EXCEPTION bit is set in sc_flags, the exception packets that caused the program check that generated in this signal are stored in sc_regs.

The definitions in the include file < machine/reg.h >), may be used to examine the exception packets. The value in sc_regs[ECR_COUNT] is the number of exception packets (zero, one or two), which are in sc_regs[EX1_CTL...EX1_RSV] and sc_regs[EX2_CTL...EX2_RSV]. Upon return from the signal handler these packets will be re-issued unless the SC_EXCEPTION bit in sc_ftags has been reset.

If the SC ONSTACK bit in sc flags is set the program was executing on the signal stack.

The SIGFPE signal (other than for integer divide) will cause the SC_FLOATSAVED bit to be set in sc_flags and floating point information to be stored in the region pointed to by sc_floatsave, the structure of this region is discussed in "Floating Point Arithmetic Linkage" in "4.3/RT Linkage Convention" in Volume II, Supplementary Documents.

The following defines the mapping of hardware traps to signals and codes. The SIGxxx symbols are defined in < signal.h>; SIGFPE codes are defined in < machine/fp.h>:

Hardware condition	Signal	Code
Program Trap	SIGTRAP	
Privileged Instruction Exception	SIGILL	mcs_pcs
Illegal Operation Code	SIGILL	mcs_pcs
Instruction or Data Address Exception:		
Page Fault (unresolvable)	SIGSEGV	
Protection	SIGSEGV	
Other	SIGBUS	
Arithmetic traps (simulated):		
Floating operation invalid	SIGFPE	FP_INV_OPER
Floating division by zero	SIGFPE	FP_DIVIDE

SIGVEC(2)

Floating underflow	SIGFPE	FP_UNDERFLOW
Floating overflow	SIGFPE	FP_OVERFLOW
Floating inexact result	SIGFPE	FP_INEXACT
Integer division by zero	SIGFPE	FP_INT_DIVIDE

BUGS

SIGVEC(2)

This manual page is confusing.

VDSPIN(2) VDSPIN(2)

NAME

```
vdspin, vdspind - spin up or spin down a Remote Virtual Disk (RVD)
```

SYNOPSIS

```
#include < machineio/vdreg.h >
#include < netinet/in.h >
vdspin(drive, args, mode, server, err_p)
unsigned long drive;
struct spinargs *args;
unsigned long mode;
struct sockaddr_in *server;
unsigned long *err_p;
vdspind(drive)
unsigned long drive;
```

DESCRIPTION

Vdspin will cause an RVD server machine to spin up a virtual disk pack. This pack must already be defined in the RVD configuration of the server machine. After being spun up, the pack is usually mounted on the client system to access its files (see mount(2) and mount(8)).

Drive is a single digit integer denoting the drive number on which to spin up the virtual disk pack. Args stores the pack name (i.e., "usr") and the optional pack password (under the misnomer of "capability") in the spinargs structure defined in < machineio/vdreg.h >:

Mode is a flag for the opening mode; its legal values are 1 for read-only, 2 for shared read/write, or 4 for exclusive read/write. Server is an internet socket address structure. Err_p is used to return error messages more specific than the system error messages returned (see below).

Warning: While an RVD pack may be spun up in shared mode, this mode is not supported; thus the results of shared access are undetermined (and files might be corrupted).

Vdspind will spin down a virtual disk pack spun up with vdspin. Drive is a single digit integer denoting the drive number to spin down.

Warning: Virtual disk packs should always be unmounted before using *vdspind*. Calling *vdspind* on a mounted RVD device may cause the client system to hang or crash when an attempt is made to access that device (see *mount*(2) and *mount*(8)).

RETURN VALUE

Vdspin: Upon successful completion, a value of 0 is returned. A value of -1 is returned in the event of failures caused by bad drive number, bad mode, or an attempt to spin up a virtual disk pack which is already spun up (from this client) on the server machine.

Vdspind: Upon successful completion, a value of 0 is returned. A value of -1 is returned in the event of failures caused by bad drive number or a non-existent disk pack.

ERRORS

Vdspin returns user error codes through err p, and one catch-all system error:

[EVDBAD] The drive does not exist, the access mode is bad, or the drive is already spun up. *Vdspind* returns only the following error:

[ENXIO] The disk pack does not exist or is not loaded, or the drive number is invalid.

VDSPIN(2)

The user error codes returned through err_p are:

[RVDEND] The drive does not exist.

[RVDEBMD] The access mode is bad.

[RVDESPN] The drive is already spun up.

SEE ALSO

down(1), up(1), spindown(8), spinup(8)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

BUGS

There are hazards associated with the access modes in *vdspin* and with the use of *vdspind*; see Warnings above.

The error codes are in a state of disarray; to be consistent with *vdspin*, *vdspind* should return the [EVDBAD] error code in the case of an invalid drive number.

VDSTATS(2)

```
NAME
vdstats - acquire client Remote Virtual Disk (RVD) statistics

SYNOPSIS
#include < netinet/in.h >
#include < machineio/vdreg.h >
vdstats(stats, drives)
struct vd_stat *stats;
struct vd_device drives[];
```

DESCRIPTION

Vdstats will acquire statistics about current virtual disk connections of this client. These statistics can be printed by using the vdstats utility (see vdstats(8)). Vdstats works by copying the global kernel variables "vdstat" and "vddinfo" into stats and drives, respectively. Stats is a pointer to a structure with fields for RVD network activity statistics, defined in < machineio/vdreg.h>:

```
struct vd stat {
           u long bad blk;
                                /* Number of bad block numbers received -
                                   i.e. responses from server that had no
                                   waiting block or drive on our side */
                                /* Number of packets with bad checksum */
           u long bad cksum;
           u long bad type;
                                /* Not a packet type that we deal with */
           u long timeout;
                                /* Number of requests that timed out */
           u long bad nonce;
                                /* Nonce did not agree */
           u long bad state;
                                /* Packet arrived; state was inappropriate */
           u long bad data;
                                /* Data was invalid */
           u long pkts sent;
                                /* Number of packets sent */
           u long blk rqs;
                                /* Number of blocks requested */
           u long rxmts;
                                /* Number of retransmits */
           u long pkts rcvd;
                                /* Number of RVD packets received */
           u long blk wrt;
                                /* Number of blocks written */
           u long q size[STQLEN];
                                /* # error pkts rcvd from server */
           u long err rcv;
           u long bad vers;
                                /* # pkts with bad version */
           u long bad frmt;
                                /* # pkts that were badly formed */
                                /* # pkts that local host wouldn't send */
           u long pkt rei:
                                /* Number of requests (bufs) delayed
           u long pushes;
                                   because of flow control */
          };
Drives is an array of vd device structures, as defined in < machineio/vdreg.h>:
        struct vd device {
          long drive;
                                /* Drive number */
                                /* Status of drive */
           u char status;
           u char bfactor;
                                /* Blocking factor for writes */
          u short state:
           u long nblocks;
                                /* Size of disk in blocks */
           u long index;
                                /* Server specified index */
           u_long nonce;
                                /* Current UID for packets from this drive */
          struct in addr server; /* Address of server where this
                                   disk actually resides */
          u short mode;
                                /* Mode spunup in */
          u short q len;
                                /* Current length of queued requests */
          u_short burst;
                                /* Maximum burst size to server */
                                /* Maximum number of outstanding requests
          u short maxqlen;
```

VDSTATS(2) VDSTATS(2)

```
on this drive */
u_short reqs_out; /* Current number of outstanding requests */
char capability[VDMAXCAPABILITY]; /* Capability for spindown */
char name[VDMAXNAME]; /* Name for vdstats */
};
```

SEE ALSO

vdstats(8)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

BUGS

The meaning of the "status" field in structure vd_device is lost.

State gives the status of the server when last accessed by the client; if there has been no activity with the server, this may become outdated.

There is no way for the user program to know how big the *drives* array should be. Since the maximum number of virtual disks is 10, the user should allocate 10 entries and zero them.

Section 3. Libraries

This section describes functions in various libraries. The functions are grouped by library. Man pages found in IBM/4.3, but not in 4.3BSD, are marked with an asterisk (*). Man pages marked with a section symbol (§) are at a level earlier than 4.3BSD.

•	intro	•	malloc
•	abort	•	math(3M)
•	asinh(3M)	•	printf(3S)
•	atof	•	scanf(3S)
•	byteorder(3N)	•	signal(3C)
•	ctype	•	sin(3M)
•	ecvt	•	sinh(3M)
•	exp(3M)	•	sqrt(3M)
•	floor(3M)	•	intro(3F)§
•	fpa(3X)*	• "	abort(3F)§
•	frexp*	•	bessel(3F)§
•	hypot(3M)	•	perror(3F)§
•	ieee*	•	rand(3F)§
•	intro(3X)	•	traper(3F)§
•	j0(3M)	•	trapov(3F)§
•	lgamma(3M)	•	trpfpe(3F)§

IBM/4.3 Manual Pages

This page intentionally left blank.

NAME

intro - introduction to C library functions

DESCRIPTION

This section describes functions that may be found in various libraries. The library functions are those other than the functions which directly invoke system primitives, described in section 2. Most of these functions are accessible from the C library, *libc*, which is automatically loaded by the C compiler cc(1). The link editor ld(1) searches this library under the '-le' option. The C library also includes all the functions described in section 2.

A subset of these functions are available from Fortran; they are described separately in intro(3F).

The functions described in this section are grouped into various sections:

- (3) The straight "3" functions are the standard C library functions.
- (3N) These functions constitute the internet network library.
- (3S) These functions constitute the 'standard I/O package', see stdio(3S) for more details. Declarations for these functions may be obtained from the include file $\langle stdio.h \rangle$.
- (3C) These routines are included for compatibility with other systems. In particular, a number of system call interfaces provided in previous releases of 4BSD have been included for source code compatibility. Use of these routines should, for the most part, be avoided. The manual page entry for each compatibility routine indicates the proper interface to use.
- (3M) These functions constitute the math library, *libm*. Functions in the math library (3M) return NAN() or INF when the function is undefined for the given arguments or when the value is not representable. In these cases the external variable *errno* (see *intro*(2)) is set to the value EDOM (domain error) or ERANGE (range error); see also the "Bugs" section in *intro*(3M). The values of EDOM and ERANGE are defined in the include file < *errno.h*>. C programs that wish to use this library need to specify the "-lm" option.
- (3X) These functions constitute minor libraries and other miscellaneous run-time facilities. Most are available only when programming in C. These functions include libraries that provide device independent plotting functions, terminal independent screen management routines for two dimensional non-bitmap display terminals, and functions for managing data bases with inverted indexes. These functions are located in separate libraries indicated in each manual entry.

FILES

/lib/libc.a the C library
/usr/lib/libm.a the math library
/usr/lib/libc_p.a the C library compiled for profiling
/usr/lib/libm p.a the math library compiled for profiling

SEE ALSO

stdio(3S), math(3M), intro(2), cc(1), ld(1), nm(1)

"Floating Point Arithmetic" in Volume II, Supplementary Documents

Appendix B, "Graphics Manual Pages" (3G), for functions supporting the Academic Information Systems experimental display.

LIST OF FUNCTIONS

OF FUNCTIONS	Annana an Dana	Description
Name	Appears on Page	Description
abort	abort.3	generate a fault
abs	abs.3	integer absolute value
acos	sin.3m	inverse trigonometric function
acosh	asinh.3m	inverse hyperbolic function
alarm	alarm.3c	schedule signal after specified time
alloca	malloc.3	memory allocator
arc	plot.3x	graphics interface
asctime	ctime.3	convert date and time to ASCII
asin	sin.3m	inverse trigonometric function
asinh	asinh.3m	inverse hyperbolic function
assert	assert.3x	program verification
atan	sin.3m	inverse trigonometric function
atanh	asinh.3m	inverse hyperbolic function
atan2	sin.3m	inverse trigonometric function
atof	atof.3	convert ASCII to numbers
atoi	atof.3	convert ASCII to numbers
atol	atof.3	convert ASCII to numbers
bcmp	bstring.3	bit and byte string operations
bcopy	bstring.3	bit and byte string operations
bzero	bstring.3	bit and byte string operations
cabs	hypot.3m	complex absolute value
calloc	malloc.3	memory allocator
cbrt	sqrt.3m	cube root
ceil	floor.3m	integer no less than
circle	plot.3x	graphics interface
clearerr	ferror.3s	stream status inquiries
closedir	directory.3	directory operations
closelog	syslog.3	control system log
closepl	plot.3x	graphics interface
cont	plot.3x	graphics interface
copysign	ieee.3m	copy sign bit
cos	sin.3m	trigonometric function
cosh	sinh.3m	hyperbolic function
crypt	crypt.3	DES encryption
ctime	ctime.3	convert date and time to ASCII
curses	curses.3x	screen functions with "optimal" cursor motion
dbminit	dbm.3x	data base subroutines
delete	dbm.3x	data base subroutines
drem	ieee.3m	remainder
ecvt	ecvt.3	output conversion
edata	end.3	last locations in program DES encryption
encrypt end	crypt.3 end.3	last locations in program
endfsent	getfsent.3x	get file system descriptor file entry
endgrent	getgrent.3	get group file entry
endhostent	gethostbyname.3n	get network host entry
endnetent	getnetent.3n	get network host entry
endprotoent	getprotoent.3n	get protocol entry
endpwent	getpwent.3	get password file entry
endservent	getservent.3n	get service entry
environ	exect.3	execute a file
VALVIII OII	OACCI.S	272220 # IIIO

erase	plot.3x	graphics interface
erf	erf.3m	error function
erfc	erf.3m	complementary error function
etext	end.3	last locations in program
exec	execl.3	execute a file
exece	exec1.3	execute a file
execl	exec1.3	execute a file
execle	execl.3	execute a file
execlp	execl.3	execute a file
exect	exec1.3	execute a file
execv	execl.3	execute a file
execvp	execl.3	execute a file
exit	exit.3	terminate a process after flushing any pending output
exp	exp.3m	exponential
expm1	exp.3m	exp(x)-1
fabs	floor.3m	absolute value
fclose	fclose.3s	close or flush a stream
fcvt	ecvt.3	output conversion
feof	ferror.3s	stream status inquiries
ferror	ferror.3s	stream status inquiries
fetch	dbm.3x	data base subroutines
fflush	fclose.3s	close or flush a stream
ffs	bstring.3	bit and byte string operations
fgetc	getc.3s	get character or word from stream
fgets	gets.3s	get a string from a stream
fileno	ferror.3s	stream status inquiries
firstkey	dbm.3x	data base subroutines
fpa	fpa.3x	direct interface to floating point accelerator
floor	floor.3m	integer no greater than
	fopen.3s	
fopen fprintf	_	open a stream
_	printf.3s	formatted output conversion
fputc	putc.3s	put character or word on a stream
fputs fread	puts.3s	put a string on a stream
_	fread.3s malloc.3	buffered binary input/output
free		memory allocator
frexp	frexp.3	split into mantissa and exponent
fscanf	scanf.3s	formatted input conversion
fseek	fseek.3s	reposition a stream
ftell	fseek.3s	reposition a stream
ftime	time.3c	get date and time
fwrite	fread.3s	buffered binary input/output
gcvt	ecvt.3	output conversion
getc	getc.3s	get character or word from stream
getchar	getc.3s	get character or word from stream
getdiskbyname	getdisk.3x	get disk description by its name
getenv	getenv.3	value for environment name
getfsent	getfsent.3x	get file system descriptor file entry
getfsfile	getfsent.3x	get file system descriptor file entry
getfsspec	getfsent.3x	get file system descriptor file entry
getfstype	getfsent.3x	get file system descriptor file entry
getgrent	getgrent.3	get group file entry
getgrgid	getgrent.3	get group file entry
getgrnam	getgrent.3	get group file entry

INTRO(3) INTRO(3)

	4141	
gethostbyaddr	gethostbyname.3n	get network host entry
gethostbyname	gethostbyname.3n	get network host entry
gethostent getlogin	gethostbyname.3n	get network host entry
getnetbyaddr	getlogin.3 getnetent.3n	get login name get network entry
getnetbyname	getnetent.3n	get network entry
getnetent	getnetent.3n	get network entry
•	_	read a password
getpass getprotobyname	getpass.3 getprotoent.3n	get protocol entry
getprotobynumber	getprotoent.3n	get protocol entry
	getprotoent.3n	get protocol entry
getprotoent getpw	getpw.3	get name from uid
getpwent	getpwent.3	get password file entry
getpwent	getpwent.3	get password file entry
getpwuid	getpwent.3	get password file entry
gets	gets.3s	get a string from a stream
_	getservent.3n	get service entry
getservbyname getservbyport	getservent.3n	get service entry
getservent	getservent.3n	get service entry
•	•	get character or word from stream
getw	getc.3s	
getwd	getwd.3 ctime.3	get current working directory pathname convert date and time to ASCII
gmtime		
gtty	stty.3c	set and get terminal state (defunct)
htonl	byteorder.3n	convert values between host and network byte order
htons	byteorder.3n	convert values between host and network byte order
hypot	hypot.3m	Euclidean distance
index	string.3	string operations
inet_addr	inet.3n	Internet address manipulation routines
inet_lnaof	inet.3n	Internet address manipulation routines
inet_makeaddr	inet.3n	Internet address manipulation routines
inet_netof	inet.3n	Internet address manipulation routines
inet_network	inet.3n	Internet address manipulation routines
infnan	infnan.3m	signals exceptions
initgroups	initgroups.3x	initialize group access list
initstate	random.3	better random number generator
insque	insque.3	insert/remove element from a queue
isalnum	ctype.3	character classification macros
isalpha 	ctype.3	character classification macros
isascii	ctype.3	character classification macros
isatty	ttyname.3	find name of a terminal
iscntrl	ctype.3	character classification macros
isdigit	ctype.3	character classification macros
islower	ctype.3	character classification macros
isprint	ctype.3	character classification macros
ispunct	ctype.3	character classification macros
isspace	ctype.3	character classification macros
isupper	ctype.3	character classification macros
j0	j0.3m	Bessel function
jl ·	j0.3m	Bessel function
jn	j0.3m	Bessel function
label	plot.3x	graphics interface
ldexp	frexp.3	split into mantissa and exponent
lgamma	lgamma.3m	log gamma function; (formerly gamma.3m)

*** ***	111 0 < 40 0	
lib2648	lib2648.3x	subroutines for the HP 2648 graphics terminal
line	plot.3x	graphics interface
linemod	plot.3x	graphics interface
localtime	ctime.3	convert date and time to ASCII
log	exp.3m	natural logarithm
logb	ieee.3m	exponent extraction
log10	exp.3m	logarithm to base 10
loglp	exp.3m	log(1+x)
longjmp malloc	setjmp.3 malloc.3	non-local goto
		memory allocator
mktemp modf	mktemp.3	make a unique file name
	frexp.3	split into mantissa and exponent
moncontrol monitor	monitor.3 monitor.3	prepare execution profile
	monitor.3	prepare execution profile
monstartup		prepare execution profile
move	plot.3x dbm.3x	graphics interface
nextkey		data base subroutines
nice	nice.3c	set program priority
nlist	nlist.3	get entries from name list
ntohl	byteorder.3n	convert values between host and network byte order
ntohs	byteorder.3n	convert values between host and network byte order
opendir	directory.3	directory operations
openlog	syslog.3	control system log
openpl	plot.3x	graphics interface
pause	pause.3c	stop until signal
pclose	popen.3	initiate I/O to/from a process
perror	perror.3	system error messages
point	plot.3x	graphics interface .
popen	popen.3	initiate I/O to/from a process
pow	exp.3m	exponential x**y
printf	printf.3s	formatted output conversion
psignal	psignal.3	system signal messages
putc	putc.3s	put character or word on a stream
putchar	putc.3s	put character or word on a stream
puts	puts.3s	put a string on a stream
putw	putc.3s	put character or word on a stream
qsort	qsort.3	quicker sort
rand	rand.3c	random number generator
random	random.3	better random number generator
rcmd	rcmd.3x	routines for returning a stream to a remote command
re_comp	regex.3	regular expression handler
re_exec	regex.3	regular expression handler
readdir	directory.3	directory operations
realloc	malloc.3	memory allocator
remque	insque.3	insert/remove element from a queue
rewind	fseek.3s	reposition a stream
rewinddir	directory.3	directory operations
rexec	rexec.3x	return stream to a remote command
rindex	string.3	string operations
rint	floor.3m	round to nearest integer
rresvport	rcmd.3x	routines for returning a stream to a remote command
ruserok	rcmd.3x	routines for returning a stream to a remote command
scalb	ieee.3m	exponent adjustment

	annudiu 2	
scandir scanf	scandir.3 scanf.3s	scan a directory
scam seekdir	· = -	formatted input conversion
setbuf	directory.3 setbuf.3s	directory operations
setbuffer	setbuf.3s	assign buffering to a stream
		assign buffering to a stream
setegid	setuid.3	set user and group ID
seteuid	setuid.3	set user and group ID
setfsent	getfsent.3x	get file system descriptor file entry
setgid	setuid.3	set user and group ID
seigieni	getgrent.3	get group file entry
sethostent	gethostbyname.3n	get network host entry
setjmp	setjmp.3	non-local goto
setkey	crypt.3	DES encryption
setlinebuf	setbuf.3s	assign buffering to a stream
setnetent	getnetent.3n	get network entry
setprotoent	getprotoent.3n	get protocol entry
setpwent	getpwent.3	get password file entry
setrgid	setuid.3	set user and group ID
setruid	setuid.3	set user and group ID
setservent	getservent.3n	get service entry
setstate	random.3	better random number generator
setuid	setuid.3	set user and group ID
signal	signal.3	simplified software signal facilities
sin	sin.3m	trigonometric function
sinh	sinh.3m	hyperbolic function
sleep	sleep.3	suspend execution for interval
space	plot.3x	graphics interface
sprintf	printf.3s	formatted output conversion
sqrt	sqrt.3m rand.3c	square root
srand		random number generator
srandom	random.3	better random number generator
sscanf	scanf.3s	formatted input conversion
stdio	intro.3s	standard buffered input/output package
store	dbm.3x	data base subroutines
strcat	string.3	string operations
stremp	string.3	string operations
strcpy	string.3	string operations
strlen	string.3	string operations
strncat	string.3	string operations
strncmp	string.3	string operations
strncpy	string.3	string operations
stty swab	stty.3c	set and get terminal state (defunct)
	swab.3	swap bytes
sys_errlist	perror.3 perror.3	system error messages
sys_nerr sys_siglist	-	system error messages system signal messages
syslog	psignal.3	control system log
	syslog.3	issue a shell command
system tan	system.3 sin.3m	trigonometric function
tan	sinh.3m	hyperbolic function
telldir	directory.3	directory operations
tgetent	termcap.3x	terminal independent operation routines
tgetflag	termcap.3x	terminal independent operation routines
Pomas	willoup.JA	comma independent operation routiles

tgetnum	termcap.3x	terminal independent operation routines
tgetstr	termcap.3x	terminal independent operation routines
tgoto	termcap.3x	terminal independent operation routines
time	time.3c	get date and time
times	times.3c	get process times
timezone	ctime.3	convert date and time to ASCII
tputs	termcap.3x	terminal independent operation routines
ttyname	ttyname.3	find name of a terminal
ttyslot	ttyname.3	find name of a terminal
ungetc	ungetc.3s	push character back into input stream
utime	utime.3c	set file times
valloc	valloc.3	aligned memory allocator
varargs	varargs.3	variable argument list
vlimit	vlimit.3c	control maximum system resource consumption
vtimes	vtimes.3c	get information about resource utilization
y0	j0.3m	Bessel function
y1	j0.3m	Bessel function
yn	j0.3m	Bessel function

This page intentionally left blank.

ABORT(3)

NAME

abort - generate a fault

SYNOPSIS

abort()

DESCRIPTION

Abort executes an instruction which is illegal in user mode. This causes a signal that normally terminates the process with a core dump, which may be used for debugging.

SEE ALSO

adb(1), sigvec(2), exit(2)

DIAGNOSTICS

Trace/BPT trap (core dumped). (From the shell)

BUGS

The abort() function does not flush standard I/O buffers. Use fflush(3S).

This page intentionally left blank.

ASINH(3M) ASINH(3M)

NAME

asinh, acosh, atanh - inverse hyperbolic functions

SYNOPSIS

```
#include < math.h >
double asinh(x)
double x;
double acosh(x)
double x;
```

double atanh(x)
double x;

DESCRIPTION

These functions compute the designated inverse hyperbolic functions for real arguments.

ERRORS (due to roundoff, etc)

These functions inherit much of their error from loglp() described in exp(3M).

DIAGNOSTICS

Acosh returns a NaN if the argument is less than 1. Atanh returns a NaN if the argument has absolute value bigger than 1, and returns $sign(x)^* \infty$ if it has an absolute value equal to 1.

SEE ALSO

 $\exp(3M)$, math(3M)

AUTHORS

W. Kahan, Kwok-Choi Ng

This page intentionally left blank.

ATOF(3)

```
NAME
atof, atoi, atol — convert ASCII to numbers

SYNOPSIS
double atof(nptr)
char *nptr;
atoi(nptr)
char *nptr;
long atol(nptr)
char *nptr;
```

DESCRIPTION

These functions convert a string pointed to by *nptr* to IEEE double, to integer, and to long integer representation, respectively. The first unrecognized character ends the string.

Atof recognizes an optional string of spaces, then an optional sign, then a string of digits optionally containing a decimal point, and then an optional "e" or "E" followed by an optionally signed integer. Conversion is rounded, with a worst-case error of 0.513 ulps (units in the last place). Values less than 2.225e-308 in magnitude have denormal representation. Values greater than 1.7976931348623158e308 in magnitude are represented by a signed infinity. Atof also accepts the strings "INF" and "NAN()", ignoring case.

Atoi and atol recognize an optional string of spaces, then an optional sign, and then a string of digits.

SEE ALSO

scanf(3S)

NAME

htonl, htons, ntohl, ntohs - convert values between host and network byte order

SYNOPSIS

```
#include < sys/types.h >
#include < netinet/in.h >
netlong = htonl(hostlong);
u_long netlong, hostlong;
netshort = htons(hostshort);
u_short netshort, hostshort;
hostlong = ntohl(netlong);
u_long hostlong, netlong;
hostshort = ntohs(netshort);
u_short hostshort, netshort;
```

DESCRIPTION

These routines convert 16- and 32-bit quantities between network byte order and host byte order. On machines such as the IBM RT PC, these routines are defined as null macros in the include file < netinet/in.h >.

These routines are most often used with Internet addresses and ports as returned by gethostent(3N) and getservent(3N).

SEE ALSO

```
gethostent(3N), getservent(3N)
```

CTYPE(3)

NAME

isalpha, isupper, islower, isdigit, isxdigit, isalnum, isspace, ispunct, isprint, isgraph, iscntrl, isascii, toupper, tolower, toascii — character classification macros

SYNOPSIS

```
#include < ctype.h >
isalpha(c)
```

DESCRIPTION

. . .

These macros classify ASCII-coded integer values by table lookup. Each is a predicate returning nonzero for true, zero for false. *Isascii* and *toascii* are defined on all integer values; the rest are defined only where *isascii* is true and on the single non-ASCII value EOF (see *stdio*(3S)).

isalpha c is a letter

isupper c is an upper case letter islower c is a lower case letter

isdigit c is a digit isxdigit c is a hex digit

isalnum c is an alphanumeric character

isspace c is a space, tab, carriage return, newline, vertical tab, or formfeed

ispunct c is a punctuation character (neither control nor alphanumeric nor space)

isprint c is a printing character, code 040(8) (space) through 0176 (tilde) isgraph c is a printing character, similar to isprint except false for space.

iscntrl c is a delete character (0177) or ordinary control character (less than 040).

isascii c is an ASCII character, code less than 0200

tolower c is converted to lower case. Return value is undefined if not isupper(c).

toupper c is converted to upper case. Return value is undefined if not islower(c).

toascii cis converted to be a valid ascii character.

SEE ALSO

ascii(7)

```
NAME
```

ecvt, fcvt, gcvt - output conversion

SYNOPSIS

char *ecvt(value, ndigit, decpt, sign)
double value;
int ndigit, *decpt, *sign;
char *fcvt(value, ndigit, decpt, sign)
double value;
int ndigit, *decpt, *sign;
char *gcvt(value, ndigit, buf)
double value;
char *buf;
#include < ieee.h >
int cvtrounding (class)
int class;

DESCRIPTION

Ecvt converts value to a null-terminated string of up to ndigit ASCII digits and returns a pointer thereto. The position of the units digit relative to the beginning of the string is stored indirectly through decpt (zero or negative means to the left of the returned digits). If the sign of value is negative, the word pointed to by sign is set non-zero; otherwise, it is zero. The decimal string is rounded in the position specified by ndigit if this represents less than the intrinsic precision of value as a type double object; otherwise, the result is one of the decimal strings of no more than 17 digits to which value is the closest floating-point approximation.

Fcvt is identical to ecvt, except that the correct digit has been rounded for FORTRAN F-format output of the number of digits specified by ndigit.

Gevt converts the value to a null-terminated ASCII string in buf and returns a pointer to buf. It attempts to produce ndigit significant digits in FORTRAN F-format, if possible, otherwise, in E-format, ready for printing. Trailing zeros may be suppressed.

If value is infinite, the string produced is "INF". If value is a NaN (Not-a-Number), the string produced is "NAN()". No distinction is made between signaling and quiet NaNs.

The cvtrounding() function determines which of two major classes of rounding is in effect. The first class is CVT_IEEE, or rounding in accordance with the IEEE standard 754 for binary floating point arithmetic. This gives four modes (defined in <ieee.h>):

TONEAREST UPWARD DOWNWARD TOWARDZERO

The particular one of these in effect is governed by the use of the rounding functions (see *ieee*(3)). The default mode is TONEAREST.

The second *class* is CVT_ROUND (rounding "to roundest"). In this, the roundest of the set of external values which map to the same internal value is chosen, and then rounded to the precision required. This is the default.

EXAMPLES

The difference between the IEEE modes and "to roundest" may be shown by several examples:

The internal value 3fb1 eb85 leb8 5lec, to 17 digits, is given as:

0.07000000000000000 CVT IEEE, TONEAREST

ECVT(3) ECVT(3)

0.07000000000000000 CVT_ROUND

The negative of this value, bfb1 eb85 leb8 5lec, to 17 digits, is given as:

-0.07000000000000000 CVT_ROUND

The internal value 3fc6 6666 6666, to 17 digits, is given as:

0.17499999999999999999 CVT IEEE, TONEAREST

0.1750000000000000 CVT ROUND

The same value, to 2 digits, is given as:

0.17

CVT_IEEE, TONEAREST

0.18

CVT_ROUND

SEE ALSO

ieee(3), printf(3)

BUGS

The return values point to static data, the content of which is overwritten by each call.

EXP(3M)EXP(3M)

```
NAME
```

exp, expm1, log, log10, log1p, pow - exponential, logarithm, power

SYNOPSIS

```
#include < math.h >
double exp(x)
double x;
double expm1(x)
double x;
double log(x)
double x;
double log 10(x)
double x;
double log1p(x)
double x;
double pow(x, y)
```

DESCRIPTION

Exp returns the exponential function of x.

Expm1 returns exp(x) - 1 accurately (even if x is close to 0).

Log returns the natural logarithm of x.

Log 10 returns the base 10 logarithm of x.

Log lp returns log(1+x) accurately (even if x is close to 0).

Pow returns x^y .

double x, y;

NOTES

Pow defines $x^0 = 1$ for all x, including zero, ∞ and NaN. Previous implementations of pow may have defined x^0 to be undefined in some or all of those cases. The reasons for setting $x^0 = 1$ in all cases are these:

- Any program that already tests whether x is 0 (or ∞ or NaN) before computing x^0 will (1)be indifferent to whether $\theta^0 = 1$ or not. Any program that expects θ^0 to be invalid is dubious anyway since that expression's meaning and, if valid, its consequences vary from one computer system to another.
- Some Algebra texts (e.g. Sigler's) define $x^0 = 1$ for all x, including x = 0. This is compa-(2)tible with the convention for polynomials that accepts $p(x) = a_0 * x^0 + a_1 * x^1 + ... + a_n * x^n$ and evaluates $p(0) = a_0$ rather than reject $a_0 * 0^0$ as invalid.

- Analysts will accept $0^0 = 1$ despite that x^y can approach anything or nothing as x and y (3) approach zero independently. The reason for setting $0^0 = 1$ anyway is this: If x(z) and y(z) are any functions analytic (expressible as power series) in z at z = 0, and if x(0) = y(0) = 0, then $x(z)^{y(z)} \rightarrow 1$ as $z \rightarrow 0$.
- If $0^0 = 1$, then $\infty^0 = 1/0^0 = 1$ too, and then $NaN^0 = 1$ because $x^0 = 1$ for all finite (4) and infinite x.

DIAGNOSTICS

Exp, expml and pow return a NaN when the correct value would overflow; errno is set to ERANGE. Pow returns a NaN and sets errno to EDOM when the first argument is negative and the second is non-integral.

Log returns a NaN when x is negative and $-\infty$ when x is zero; errno is set to EDOM.

ERRORS

Exp, log, expm and loglp are accurate to within an ulp and log10 to within about 2 ulps; an ulp is one Unit in the Last Place. The error in pow is below about 2 ulps when its magnitude is moderate, but increases as pow approaches the over/underflow thresholds until as many bits could be lost as are occupied by the floating-point format's exponent field (11 bits for double precision). No such drastic loss has been exposed by testing; the worst errors observed have been below 300 ulps. Moderate values of pow are accurate enough that pow(integer,integer) is exact until it is bigger than 2**53.

SEE ALSO

math(3M)

AUTHORS

Kwok-Choi Ng, W. Kahan

FLOOR(3M) FLOOR(3M)

```
NAME
        fabs, floor, ceil - absolute value, floor, ceiling functions
SYNOPSIS
        #include < math.h >
        double floor(x)
        double x;
        double ceil(x)
        double x;
        double fabs(x)
        double x;
DESCRIPTION
        Fabs returns the absolute value |x|.
        Floor returns the largest integer not greater than x.
        Ceil returns the smallest integer not less than x.
SEE ALSO
        abs(3), ieee(3M), math(3M)
```

FPA(3X)

NAME

fpa - direct interface to floating point accelerator

DESCRIPTION

This library is no longer required for 4.3/RT floating point support. However, it is provided in this release to provide compatibility with previous releases and to enable an orderly conversion to the new support. Users are encouraged to recompile all floating point programs. 4.3/RT floating point support has been enhanced so that it will use the fastest floating point hardware available (unless use of the FPA environment variable specifies otherwise). For more information, see the two articles mentioned below.

For those systems where users have many Makefiles, scripts, and so forth, that depend on the -lfpa flag, the system administrator can install a dummy library to satisfy the loader. A dummy library is provided for this purpose in \(\lambda usr/src/old/fpa. \)

SEE ALSO

"IBM/4.3 Linkage Convention" in Volume II, Supplementary Documents

"Floating Point Arithmetic" in Volume II, Supplementary Documents

FREXP(3)

NAME

frexp, ldexp, modf - split into mantissa and exponent

SYNOPSIS

```
double frexp(value, eptr)
double value;
int *eptr;
double ldexp(value, exp)
double value, exp;
double modf(value, iptr)
double value, *iptr;
```

DESCRIPTION

Frexp returns the mantissa of value as a double quantity, x, of magnitude less than 1; x is less than 0.5 only if value is denormal. Frexp also stores in *eptr an integer n such that value = $x*2^n$.

Ldexp returns the quantity value * 2^{exp} by calling scalb with the same arguments.

Modf returns the fractional part of value and stores the integer part in *iptr.

DIAGNOSTICS

Frexp returns sign(x) * 0.5 and stores 50000 in *eptr when x is infinite; it returns a NaN and stores 0 in *eptr when x is a NaN, and returns 0 and stores 0 in *eptr when x is 0.

Ldexp sets errno to ERANGE on overflow or underflow.

Modf returns 0 and stores x in *iptr when x is infinite; it returns x and stores x in *iptr when x is a NaN.

SEE ALSO

ieee(3)

LGAMMA(3M)

LGAMMA(3M)

NAME

lgamma - log gamma function

SYNOPSIS

```
#include < math.h >
double lgamma(x)
double x;
```

DESCRIPTION

```
Lgamma returns \ln |\Gamma(x)| where \Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt for x > 0 and \Gamma(x) = \pi/(\Gamma(1-x)\sin(\pi x)) for x < 1.
```

The sign of $\Gamma(x)$ is returned in the external integer signgam. Do not use the expression

```
g = \exp(\operatorname{lgamma}(x)) * \operatorname{signgam}
```

to compute $g = \Gamma(x)$. Instead, use a sequence of statements such as

```
lg = lgamma(x);
g = exp(lg)*signgam;
```

 $g = (g = \exp(lgamma(x)), g*signgam);$

because only after lgamma has returned can the value of signgam be relied on. Note, too, that evaluation of $\Gamma(x)$ must overflow when x is large enough, underflow when -x is large enough, and divide by zero when x is a nonpositive integer.

NOTES

In 4.2BSD and earlier systems, the log gamma function is named gamma. The name has been changed to *lgamma* to correspond to mathematical reality.

The original C's gamma probably delivered $\ln(\Gamma(|x|))$. Later, the program gamma was changed to cope with negative arguments x in a more conventional way, but the documentation did not reflect that change correctly. The most recent change corrects inaccurate values when x is almost a negative integer, and lets $\Gamma(x)$ be computed without conditional expressions. Programmers should not assume that lgamma has settled down.

Programmers who have to use the name gamma in its former sense, for what is now lgamma, can add the following program to their others:

```
#include < math.h >
double gamma(x)
double x;
{
    return (lgamma(x));
}
```

DIAGNOSTICS

For nonpositive integral arguments, lgamma returns ∞ and errno is set to EDOM. For huge arguments, underflow or overflow may be signaled.

SEE ALSO

math(3M)

HYPOT(3M)

NAME

hypot, cabs - Euclidean distance, complex absolute value

SYNOPSIS

```
#include < math.h >
double hypot(x, y)
double x, y;
double cabs(z)
struct { double x, y;} z;
```

DESCRIPTION

Hypot(x,y) and cabs(x,y) return sqrt(x*x+y*y) computed in such a way that underflow will not happen, and overflow occurs only if the final result deserves it.

```
hypot(\infty, v) = hypot(v, \infty) = + \infty for all v including NaN.
```

NOTES

As might be expected, hypot(v,NaN) and hypot(NaN,v) are NaN for all finite v. Programmers might be surprised at first to discover that $hypot(\pm \infty, NaN) = +\infty$. This is intentional; it happens because $hypot(\infty,v) = +\infty$ for all v, finite or infinite. Hence $hypot(\infty,v)$ is independent of v. NaN is designed to disappear when it turns out to be irrelevant, as it does in $hypot(\infty,NaN)$.

ERRORS (due to roundoff, etc.)

Below 0.97 *ulps*. Consequently hypot(5.0,12.0) = 13.0 exactly; in general, hypot and *cabs* return an integer whenever an integer might be expected.

SEE ALSO

math(3M), sqrt(3M)

AUTHOR

W. Kahan

IEEE(3)

NAME

copysign, drem, logb, scalb, rint, classdouble, classfloat, isnan, unordered, finite, infinity, nextdouble, nextfloat, fptestround, fpsetround, swapround, fptestflag, fpsetflag, fpclrflag, swapfpflag, fptest-trap, fpsettrap, swapfptrap — ieee arithmetic support functions

SYNOPSIS

```
#include < ieee.h >
double copysign(x, y)
double x, y;
double drem(x, y)
double x, y;
double logb(x)
double x;
double scalb(x, n)
double x;
int n;
double rint(x)
double x;
NUMCLASS classdouble(x)
double x;
NUMCLASS classfloat(x)
double x;
int isnan(x)
double x;
int unordered(x, y)
double x, y;
int finite(x)
double x;
double infinity();
double nextdouble(x, y)
double x, y;
float nextfloat(x, y)
double x, y;
ROUNDDIR fptestround();
void fpsetround(r)
ROUNDDIR r;
ROUNDDIR swapround(r)
ROUNDDIR r;
FPEXCEPTION fptestflag();
void fpsetflag (m)
FPEXCEPTION m;
void fpclrflag(m)
FPEXCEPTION m;
FPEXCEPTION swapfpflag(m, e)
FPEXCEPTION m, e;
```

IEEE(3)

IEEE(3)

FPEXCEPTION fptesttrap();

void fpsettrap(m)
FPEXCEPTION m:

void fpclrtrap(m)

FPEXCEPTION m;

FPEXCEPTION swapfptrap(m, e)

FPEXCEPTION m, e;

DESCRIPTION

These functions are necessary for, or recommended by, the IEEE standard 754 for binary floating-point arithmetic. They observe the rounding mode and set exception flags as appropriate.

Copysign returns x with the sign of y.

Drem returns r when y is not equal to 0; the remainder r is defined by the mathematical relation

$$r = x - y * n$$

where n is the integer nearest the exact value x/y; whenever |n-x/y| = 1/2, then n is even. Consequently the remainder is computed exactly and $|r| \le |y|/2$. But drem(x,0) is exceptional; see below under "Diagnostics".

Logb returns the exponent of x, a signed integer represented as a double precision number. Here the exponent of x is defined to be the integer n such that $1 < = |x/(2^n)| < 2$ unless x = 0 or $|x| = \infty$ or x lies between 0 and the Underslow Threshold; see "Bugs". If the magnitude of x is ∞ , $\log b$ returns ∞ .

Scalb returns $x^*(2^n)$, for integer values n without computing 2^n .

Rint returns x rounded to an integral value, according to the current rounding mode. In the default rounding mode, round-to-nearest, rint(x) is the integer nearest x with the additional stipulation that |rint(x) - x| = 1/2, then rint(x) is even. Other rounding modes can make rint act like floor, or like ceil, or round toward zero.

Classdouble and classfloat return the numerical class of x, which is either signaling NaN (SNAN), quiet NaN (QNAN), infinity (INFINITE), zero (ZERONUM), denormal (DENORMALNUM), or normal (NORMALNUM).

Isnan returns a value of 1 if x is a NaN (not-a-number); otherwise, it returns 0.

Unordered returns a value of 1 if x is unordered with y (i.e. x or y is a NaN); otherwise, it returns 0.

Finite returns a value of 1 if $-\infty < x < \infty$; otherwise, it returns 0.

Infinity returns the value ∞ .

Nextdouble and nextfloat return the next representable neighbor of x in the direction toward y.

Fptestround returns the current rounding mode.

Fpsetround sets the rounding mode to r.

Swapround sets the rounding mode to r and returns the previous rounding mode.

Fptestflag returns the current status of all exception flags.

Fpsetflag sets the exception flags marked in mask m.

Fpclrflag clears the exception flags marked in mask m.

Swapfpflag sets each exception flag marked in mask m to the corresponding setting (1 = set, 0 = clear) in e, and returns the previous status of each flag. If the mask is 0, no flag is changed and the current status of each flag marked in e is returned.

Fptesttrap returns the current setting of all exception traps.

Fpsettrap enables traps for the exceptions marked in mask m.

Fpclrtrap disables traps for the exceptions marked in mask m.

Swapfptrap sets each exception trap marked in the mask m to the corresponding setting (1 = enables, 0 = disables) in e, and returns the previous setting of each exception trap. If the mask is 0, no exception trap is changed and the current setting of each exception trap marked in e is returned.

```
< ieee.h > has the following form:
* 5799-CGZ (C) COPYRIGHT IBM CORPORATION 1986,1987
* LICENSED MATERIALS - PROPERTY OF IBM
* REFER TO COPYRIGHT INSTRUCTIONS FORM NUMBER G120-2083
/* ieee.h */
/* types for recommended functions and support routines
typedef enum {
                                /* allows user setting and
      TONEAREST,
                                                                   */
                                       /* clearing of rounding mode
      UPWARD.
      DOWNWARD.
      TOWARDZERO
} ROUNDDIR;
typedef enum {
                                /* every ieee floating point
      SNAN,
                                /* value is in one and only one
      ONAN.
                                /* of these classes
      INFINITE,
      ZERONUM,
      NORMALNUM,
      DENORMALNUM
} NUMCLASS;
#define FPINVALID 1
                                /* allows user to set and clear
                          2
#define FPUNDERFLOW
                                      /* floating point exception flags */
#define FPOVERFLOW
                                      /* and traps
#define FPDIVBYZERO
#define FPINEXACT 16
#define FPALLEXCEPTIONS 0x1f
typedef short FPEXCEPTION;
                                /* a sum of particular exceptions */
  IEEE recommended functions and recommended support routines.
double copysign(), rint(), scalb(), logb(), drem(), nextdouble(), infinity();
float nextfloat();
NUMCLASS classfloat(), classdouble();
```

DIAGNOSTICS

Drem returns a NaN when y is equal to 0 or x is infinite; in both cases an invalid operation exception is raised.

Logb returns $-\infty$ when x is equal to 0, and signals the division by zero exception.

Scalb sets errno to ERANGE on overflow.

NOTES

The IEEE 754 Standard for Binary Floating Point Arithmetic can be obtained from:

IEEE Standards Office 345 East 47th Street New York, NY 10017

BUGS

Classfloat will never return SNAN. (All float arguments get converted to double before being passed to a routine; thus, a signaling NaN, upon conversion, will become quiet before classfloat receives it.)

IEEE 754 currently specifies that logb (denormalized no.) = logb (tiniest normalized no. > 0) but the consensus has changed to the specification in the new IEEE standard 854, namely that logb(x) satisfy

 $1 \le \text{scalb}(|\mathbf{x}|, -\log b(\mathbf{x})) < \text{Radix} \quad (\text{Radix} = 2 \text{ for IEEE 754})$

for every x except 0, ∞ and NaN. Almost every program that assumes 754's specification will work correctly if logb follows 854's specification instead.

SEE ALSO

math(3M)

INTRO(3X)

NAME

intro - introduction to miscellaneous library functions

DESCRIPTION

These functions constitute minor libraries and other miscellaneous run-time facilities. Most are available only when programming in C. The list below includes libraries which provide device independent plotting functions, terminal independent screen management routines for two dimensional non-bitmap display terminals, functions for managing data bases with inverted indexes, and sundry routines used in executing commands on remote machines. The routines getdiskbyname, rcmd, rresvport, ruserok, and rexec reside in the standard C run-time library "-lc". All other functions are located in separate libraries indicated in each manual entry.

FILES

/lib/libc.a /usr/lib/libdbm.a /usr/lib/libtermcap.a /usr/lib/libcurses.a /usr/lib/lib2648.a /usr/lib/libplot.a /usr/lib/libfpa.a

LIST OF FUNCTIONS

Name	Appears on Page	e Description
arc	plot.3x	graphics interface
assert	assert.3x	program verification
circle	plot.3x	graphics interface
closepl	plot.3x	graphics interface
cont	plot.3x	graphics interface
curses	curses.3x	screen functions with "optimal" cursor motion
dbminit	dbm.3x	data base subroutines
delete	dbm.3x	data base subroutines
endfsent	getfsent.3x	get file system descriptor file entry
erase	plot.3x	graphics interface
fetch	dbm.3x	data base subroutines
firstkey	dbm.3x	data base subroutines
fpa	fpa.3x	direct interface to floating point accelerator
getdiskbyname	getdisk.3x	get disk description by its name
getfsent	getfsent.3x	get file system descriptor file entry
getfsfile	getfsent.3x	get file system descriptor file entry
getfsspec	getfsent.3x	get file system descriptor file entry
getfstype	getfsent.3x	get file system descriptor file entry
initgroups	initgroups.3x	initialize group access list
label	plot.3x	graphics interface
lib2648	lib2648.3x	subroutines for the HP 2648 graphics terminal
line	plot.3x	graphics interface
linemod	plot.3x	graphics interface
move	plot.3x	graphics interface
nextkey	dbm.3x	data base subroutines
openpl	plot.3x	graphics interface
point	plot.3x	graphics interface
rcmd	rcmd.3x	routines for returning a stream to a remote command
rexec	rexec.3x	return stream to a remote command
rresvport	rcmd.3x	routines for returning a stream to a remote command
ruserok	rcmd.3x	routines for returning a stream to a remote command
setfsent	getfsent.3x	get file system descriptor file entry

INTRO(3X) INTRO(3X)

space	plot.3x	graphics interface
store	dbm.3x	data base subroutines
tgetent	termcap.3x	terminal independent operation routines
tgetflag	termcap.3x	terminal independent operation routines
tgetnum	termcap.3x	terminal independent operation routines
tgetstr	termcap.3x	terminal independent operation routines
tgoto	termcap.3x	terminal independent operation routines
tputs	termcap.3x	terminal independent operation routines

J0(3M)

DIAGNOSTICS

```
NAME
        j0, j1, jn, y0, y1, yn - Bessel functions
SYNOPSIS
        #include < math.h >
        double j0(x)
        double x;
        double j1(x)
        double x;
        double jn(n, x)
        int n;
        double x;
        double y0(x)
        double x;
        double y1(x)
        double x;
        double yn(n, x)
        int n;
        double x;
DESCRIPTION
        These functions calculate Bessel functions of the first and second kinds for real arguments and
       integer orders.
SEE ALSO
       math(3M)
```

Negative arguments cause y0, y1, and yn to return $-\infty$ and set errno to EDOM.

LGAMMA(3M)

LGAMMA(3M)

```
NAME
```

lgamma - log gamma function

SYNOPSIS

```
#include < math.h >
double lgamma(x)
double x;
```

DESCRIPTION

```
Lgamma returns \ln |\Gamma(x)| where \Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt for x > 0 and \Gamma(x) = \pi/(\Gamma(1-x)\sin(\pi x)) for x < 1.
```

The sign of $\Gamma(x)$ is returned in the external integer signgam. Do not use the expression $g = \exp(\lg \operatorname{amma}(x)) * \operatorname{signgam}$

to compute $g = \Gamma(x)$. Instead, use a sequence of statements such as

```
lg = lgamma(x);
g = exp(lg)*signgam
```

 $g = \exp(\lg) * signgam;$

 $g = (g = \exp(lgamma(x)), g*signgam);$

because only after lgamma has returned can the value of signgam be relied on. Note, too, that evaluation of $\Gamma(x)$ must overflow when x is large enough, underflow when -x is large enough, and divide by zero when x is a nonpositive integer.

NOTES

In 4.2BSD and earlier systems, the log gamma function is named gamma. The name has been changed to lgamma to correspond to mathematical reality.

The original C's gamma probably delivered $\ln(\Gamma(|x|))$. Later, the program gamma was changed to cope with negative arguments x in a more conventional way, but the documentation did not reflect that change correctly. The most recent change corrects inaccurate values when x is almost a negative integer, and lets $\Gamma(x)$ be computed without conditional expressions. Programmers should not assume that lgamma has settled down.

Programmers who have to use the name gamma in its former sense, for what is now lgamma, can add the following program to their others:

```
#include < math.h >
double gamma(x)
double x;
{
    return (lgamma(x));
}
```

DIAGNOSTICS

For nonpositive integral arguments, lgamma returns ∞ and errno is set to EDOM. For huge arguments, underflow or overflow may be signaled.

SEE ALSO

math(3M)

MALLOC(3)

NAME

malloc, free, realloc, calloc, alloca - memory allocator

SYNOPSIS

char *malloc(size)
unsigned size;
free(ptr)
char *ptr;
char *realloc(ptr, size)
char *ptr;
unsigned size;
char *calloc(nelem, elsize)
unsigned nelem, elsize;
char *alloca(size)

DESCRIPTION

int size;

Malloc and free provide a general-purpose memory allocation package. Malloc returns a pointer to a block of at least size bytes beginning on a word boundary.

The argument to *free* is a pointer to a block previously allocated by *malloc*; this space is made available for further allocation, but its contents are left undisturbed.

Needless to say, grave disorder will result if the space assigned by malloc is overrun or if some random number is handed to free.

Malloc maintains multiple lists of free blocks according to size, allocating space from the appropriate list. It calls sbrk (see brk(2)) to get more memory from the system when there is no suitable space already free.

Realloc changes the size of the block pointed to by ptr to size bytes and returns a pointer to the (possibly moved) block. The contents will be unchanged up to the lesser of the new and old sizes.

In order to be compatible with older versions, realloc also works if ptr points to a block freed since the last call of malloc, realloc or calloc; sequences of free, malloc and realloc were previously used to attempt storage compaction. This procedure is no longer recommended.

Calloc allocates space for an array of nelem elements of size elsize. The space is initialized to zeros.

Alloca allocates size bytes of space in the stack frame of the caller. This temporary space is automatically freed on return. The functions that call alloca must be compiled with the -ma flag (see hc(1) and pcc(1)). Alloca will abort if this is not the case.

Each of the allocation routines returns a pointer to space suitably aligned (after possible pointer coercion) for storage of any type of object. If the space is of *pagesize* or larger, the memory returned will be page-aligned.

SEE ALSO

brk(2), pagesize(2)

"IBM/4.3 Linkage Convention" in Volume II, Supplementary Documents, for alloca

DIAGNOSTICS

Malloc, realloc and calloc return a null pointer (0) if there is no available memory or if the arena has been detectably corrupted by storing outside the bounds of a block. Malloc may be recompiled to check the arena very stringently on every transaction; those sites with a source code license may check the source code to see how this can be done.

MALLOC(3) MALLOC(3)

Alloca returns a null pointer if an illegal size (less than or equal to 0 bytes) is requested.

BUGS

When realloc returns 0, the block pointed to by ptr may be destroyed.

The current implementation of *malloc* does not always fail gracefully when system memory limits are approached. It may fail to allocate memory when larger free blocks could be broken up, or when limits are exceeded because the size is rounded up. It is optimized for sizes that are powers of two.

Alloca is machine dependent; its use is discouraged.

MATH(3M) MATH(3M)

NAME

math - introduction to mathematical library functions

DESCRIPTION

These functions constitute the math library, *libm*. They are automatically loaded as needed by the FORTRAN compiler f77(1). The link editor searches this library under the -lm option. Declarations for these functions may be obtained from the include file < math.h >.

LIST OF FUNCTIONS

Name	Appears on Page	Description
acos	sin.3m	inverse trigonometric functions
acosh	asinh.3m	inverse hyperbolic functions
asin	sin.3m	inverse trigonometric functions
asinh	asinh.3m	inverse hyperbolic functions
atan	sin.3m	inverse trigonometric functions
atanh	asinh.3m	inverse hyperbolic functions
atan2	sin.3m	inverse trigonometric functions
cabs	hypot.3m	complex absolute value
cbrt	sqrt.3m	cube root
ceil	floor.3m	interger no less than
cos	sin.3m	trigonometric function
cosh	sinh.3m	hyperbolic function
erf	erf.3m	error function
erfc	erf.3m	complementary error function
exp	exp.3m	exponential
expm1	exp.3m	exp(x) - 1
fabs	floor.3m	absolute value
floor	floor.3m	integer no greater than
gamma	lgamma.3m	name changed to Igamma
hypot	hypot.3m	Euclidean distance
j0	j0.3m	bessel function
jl	j0.3m	bessel function
jn	j0.3m	bessel function
lgamma	lgamma.3m	log gamma function (formerly gamma.3m)
log	exp.3m	natural logarithm
log10	exp.3m	logarithm to base 10
loglp	exp.3m	$\log(1+x)$
pow	exp.3m	exponential x**y
sin	sin.3m	trigonometric function
sinh	sinh.3m	hyperbolic functions
sqrt	sqrt.3m	square root
tan	sin.3m	trigonometric function
tanh	sinh.3m	hyperbolic function
y0	j0.3m	bessel function
y1	j0.3m	bessel function
yn	j0.3m	bessel function

BUGS

Functions in the math library should set *errno* when the function is undefined for given arguments or when the function value is not representable; in some cases this is not done. It is done, however, for all instances mentioned in the "Diagnostics" sections of 3M man pages.

SEE ALSO

"Floating Point Arithmetic" in Volume II, Supplementary Documents.

PRINTF(3S)

PRINTF(3S)

```
NAME
        printf, fprintf, sprintf - formatted output conversion
SYNOPSIS
        #include < stdio.h >
        printf(format [, arg ] ... )
        char *format;
        fprintf(stream, format [, arg ] ... )
        FILE *stream;
        char *format;
        sprintf(s, format [, arg ] ... )
        char *s, format;
        #include < varargs.h >
        doprnt(format, args, stream)
        char *format;
        va list *args;
        FILE *stream;
```

DESCRIPTION

Printf places output on the standard output stream stdout. Fprintf places output on the named output stream. Sprintf places "output" in the string s, followed by the character '\0'. These routines work by calling the internal routine _doprnt, using the variable-length argument facilities of varargs(3).

Each of these functions converts, formats, and prints its arguments under control of the *format*. The *format* is a character string that contains two types of objects: plain characters, which are simply copied to the output stream, and conversion specifications, each of which causes conversion and printing of the next successive argument to *printf*.

Each conversion specification is introduced by the character "%". The remainder of the conversion specification includes in the following order

- Zero or more of following flags:
 - a "#" character specifying that the value should be converted to an alternate form. For c, d, s, and u, conversions, this option has no effect. For o conversions, the precision of the number is increased to force the first character of the output string to a zero. For x(X) conversion, a non-zero result has the string 0x(0X) prepended to it. For e, E, f, g, and G, conversions, the result will always contain a decimal point, even if no digits follow the point (normally, a decimal point only appears in the results of those conversions if a digit follows the decimal point). For g and G conversions, trailing zeros are not removed from the result as they would otherwise be.
 - a minus sign "-" which specifies left adjustment of the converted value in the indicated field.
 - a blank which specifies that a blank is to precede any non-negative value in d, e, E, f, g, or G conversion.
 - a plus sign "+" which specifies that a "+" is to precede any non-negative value in d, e, E, f, g, or G conversion. A "+" overrides a blank.
- an optional decimal digit string specifying a field width. If the converted value has fewer characters than the field width, it will be blank-padded on the left (or on the right, if the left-adjustment indicator has been given) to make up the field width. (The field width string may begin with an optional zero "0" to specify left-padding with zeroes rather than blanks. This option is ignored for infinity and NaN values.)

- an optional period "." which serves to separate the field width from the next digit string.
- an optional decimal digit string specifying a precision that specifies the number of digits to appear after the decimal point, for e, E, and f conversion; the minimum number of digits to appear for integer conversions; or the maximum number of characters to be printed from a string.
- the character I specifying that a following d, o, x, or u corresponds to a long integer arg. Not needed; has no effect.
- an optional character h specifying that a following d, o, x, X or u corresponds to a short integer arg. Not needed; has no effect.
- a character which indicates the type of conversion to be applied.

A field width or precision may be "*" instead of a digit string; the next arg is an integer supplying the field width or precision. Precisions must be non-negative integer values. A negative arg for field width represents both a '-' flag and a field width of |arg|. The maximum field width for both string and numeric conversions is 30,000; the maximum precision specification is 500 for floating-point conversions and 100 for integer conversions.

The conversion characters and their meanings are:

- d,o,x,X The integer arg is converted to decimal, octal, or hexadecimal notation, respectively. x uses "abcdef"; X uses "ABCDEF".
- The float or double arg is converted to decimal notation in the style "[-]ddd.ddd" where the number of d's after the decimal point is equal to the precision specification for the argument. If the precision is unspecified, 6 digits are produced; if the precision is explicitly 0, no digits and no decimal point are produced. Infinity prints as INF; NaNs print as NAN().
- e,E The float or double arg is converted in the style "[-]d.ddde±dd" where there is one digit before the decimal point and the number after is equal to the precision specification for the argument. If the precision is unspecified, six digits are produced. Infinity prints as INF; NaNs print as NAN().
- g,G The float or double value is converted to an f-format string, if appropriate, or an e- or E-format string, if not. The precision specification (if not specified, 6) designates the number of significant digits. Trailing zeros are stripped. Infinity prints as INF; NaNs print as NAN().
- c The character arg is printed.
- s Arg is taken to be a string (character pointer) and characters from the string are printed until a null character or until the number of characters indicated by the precision specification is reached; however, if the precision is 0 or unspecified, all characters up to a null or the maximum string field width are printed.
- u The unsigned integer arg is converted to decimal and printed (the result will be in the range 0 through MAXUINT, where MAXUINT equals 4294967295).
- % Print a "%"; no argument is converted.

D,O,U Obsolete equivalent to Id,Io,Iu.

A nonexistent or small field width does not cause truncation of a field; padding takes place only if the specified field width exceeds the actual width. Characters generated by *printf* are printed by *putc*(3S).

Examples

To print a date and time in the form "Sunday, July 3, 10:02", where weekday and month are pointers to null-terminated strings:

PRINTF(3S)

PRINTF(3S)

```
printf("%s, %s %d, %02d:%02d", weekday, month, day, hour, min);
```

To print π to 5 decimals:

To print a floating-point value to full precision while minimizing the number of digits:

SEE ALSO

ecvt(3), putc(3S), scanf(3S)

NAME

```
scanf, fscanf, sscanf - formatted input conversion
```

SYNOPSIS

```
#include < stdio.h >
scanf(format [ , pointer ] . . . )
char *format;
fscanf(stream, format [ , pointer ] . . . )
FILE *stream;
char *format;
sscanf(s, format [ , pointer ] . . . )
char *s, *format;
```

DESCRIPTION

Scanf reads from the standard input stream stdin. Fscanf reads from the named input stream. Sscanf reads from the character string s. Each function reads characters, interprets them according to a format, and stores the results in its arguments. Each expects as arguments a control string format, described below, and a set of pointer arguments indicating where the converted input should be stored.

The control string usually contains conversion specifications, which are used to direct interpretation of input sequences. The control string may contain:

- 1. Blanks, tabs or newlines, which match optional white space in the input.
- 2. An ordinary character (not %) which must match the next character of the input stream.
- 3. Conversion specifications, consisting of the character %, an optional assignment suppressing character *, an optional numerical maximum field width, and a conversion character.

A conversion specification directs the conversion of the next input field; the result is placed in the variable pointed to by the corresponding argument, unless assignment suppression was indicated by *. An input field is defined as a string of non-space characters; it extends to the next inappropriate character or until the field width, if specified, is exhausted.

The conversion character indicates the interpretation of the input field; the corresponding pointer argument must usually be of a restricted type. The following conversion characters are legal:

- % a single "%" is expected in the input at this point; no assignment is done.
- d a decimal integer is expected; the corresponding argument should be an integer pointer.
- o an octal integer is expected; the corresponding argument should be a integer pointer.
- x a hexadecimal integer is expected; the corresponding argument should be an integer pointer.
- s a character string is expected; the corresponding argument should be a character pointer pointing to an array of characters large enough to accept the string and a terminating "\0", which will be added. The input field is terminated by a whitespace character.
- a character is expected; the corresponding argument should be a character pointer. The normal skip over space characters is suppressed in this case; to read the next non-space character, use "%1s". If a field width is given, the corresponding argument should refer to a character array, and the indicated number of characters is read.
- e,f a floating-point number is expected; the next field is converted accordingly and stored through the corresponding argument, which should be a pointer to a *float*. The input format for floating-point numbers is one of the following: (a) an optionally signed string of digits possibly containing a decimal point, followed by an optional exponent field consisting of an E or e followed by an optionally signed integer; (b) the characters "INF" in upper or lower case optionally preceded by a sign, or (c) the characters "NAN" in upper or lower case fol-

}

SCANF(3S) SCANF(3S)

lowed by "()" and optionally preceded by a sign.

indicates a string not to be delimited by space characters. The left bracket is followed by a set of characters and a right bracket; the characters between the brackets define a set of characters making up the string. If the first character is not circumflex (^), the input field is all characters until the first character not in the set between the brackets; if the first character after the left bracket is ^, the input field is all characters until the first character which is in the remaining set of characters between the brackets. The corresponding argument must point to a character array.

The conversion characters \mathbf{d} , \mathbf{o} and \mathbf{x} may be capitalized or preceded by \mathbf{l} to indicate that a pointer to long rather than to int is in the argument list. Similarly, the conversion characters \mathbf{e} and \mathbf{f} may be capitalized or preceded by \mathbf{l} to indicate a pointer to double rather than to float. The conversion characters \mathbf{d} , \mathbf{o} and \mathbf{x} may be preceded by \mathbf{h} to indicate a pointer to short rather than to int.

The scanf functions return the number of successfully matched and assigned input items. This can be used to decide how many input items were found. The constant EOF is returned upon end of input; note that this is different from 0, which means that no conversion was done; if conversion was intended, it was frustrated by an inappropriate character in the input.

For example, the call

```
int i; float x; char name[50]; scanf("%d%f%s", &i, &x, name);
```

with the input line

```
25 54.32E-1 thompson
```

will assign to i the value 25, x the value 5.432, and name will contain "thompson\0". Or,

```
int i; float x; char name[50]; scanf("%2d%f%*d%[1234567890]", &i, &x, name);
```

with input

56789 0123 56a72

will assign 56 to i, 789.0 to x, skip "0123", and place the string "56\0" in name. The next call to getchar will return "a".

NOTE

When scanf returns on a failed conversion, the stream may have been consumed beyond the start of the input field that failed to convert. For example, after scanf("%d",&n) on the input -d, the result of getchar() is "d", not "-". After scanf("%lf",&d) on the input -intuitive, the result of getchar() is "t", not "-" or "i".

SEE ALSO

```
atof(3), getc(3S), printf(3S)
```

DIAGNOSTICS

The scanf functions return EOF on end of input, and a short count for missing or illegal data items.

BUGS

The success of literal matches and suppressed assignments is not directly determinable.

SIGNAL(3C)

```
NAME
```

signal - simplified software signal facilities

SYNOPSIS

#include < signal.h >
 (*signal(sig, func))()
int (*func)();

DESCRIPTION

Signal is a simplified interface to the more general sigvec(2) facility.

A signal is generated by some abnormal event begun by a user at a terminal (quit, interrupt, stop), by a program error (bus error, etc.), by request of another program (kill), or when a process is stopped because it wishes to access its control terminal while in the background (see tty(4)). Signals are optionally generated when a process resumes after being stopped, when the status of child processes changes, or when input is ready at the control terminal. Most signals cause termination of the receiving process if no action is taken; some signals instead cause the process receiving them to be stopped, or are simply discarded if the process has not requested otherwise. Except for the SIGKILL and SIGSTOP signals, the signal call allows signals either to be ignored or to cause an interrupt to a specified location. The following is a list of all signals with names as in the include file < signal.h >:

1	SIGHUP	hangup
2	SIGINT	interrupt
3*	SIGQUIT	quit
4*	SIGILL	illegal instruction
5*	SIGTRAP	trace trap
6*	SIGIOT	IOT instruction
7*	SIGEMT	EMT instruction
8*	SIGFPE	floating-point exception
9	SIGKILL	kill (cannot be caught or ignored)
10*	SIGBUS	bus error
11*	SIGSEGV	segmentation violation
12*	SIGSYS	bad argument to system call
13	SIGPIPE	write on a pipe with no one to read it
14	SIGALRM	alarm clock
15	SIGTERM	software termination signal
16●	SIGURG	urgent condition present on socket
17†	SIGSTOP	stop (cannot be caught or ignored)
18†	SIGTSTP	stop signal generated from keyboard
19•	SIGCONT	continue after stop
20●	SIGCHLD	child status has changed
21†	SIGTTIN	background read attempted from control terminal
22†	SIGTTOU	background write attempted to control terminal
23•	SIGIO	I/O is possible on a descriptor (see fcntl(2))
24	SIGXCPU	cpu time limit exceeded (see setrlimit(2))
25	SIGXFSZ	file size limit exceeded (see setrlimit(2))
26	SIGVTALRM	virtual time alarm (see setitimer(2))
27	SIGPROF	profiling timer alarm (see setitimer(2))
28●	SIGWINCH .	window size change
30	SIGUSR1	user-defined signal 1
31	SIGUSR2	user-defined signal 2

The starred (*) signals in the list above cause a core image if not caught or ignored.

SIGNAL(3C) SIGNAL(3C)

If func is SIG_DFL, the default action for signal sig is reinstated; this default is termination (with a core image for starred signals) except for signals marked with • or †. Signals marked with • are discarded if the action is SIG_DFL; signals marked with † cause the process to stop. If func is SIG_IGN, the signal is subsequently ignored and pending instances of the signal are discarded. Otherwise, when the signal occurs further occurrences of the signal are automatically blocked and func is called.

A return from the function unblocks the handled signal and continues the process at the point it was interrupted. Unlike previous signal facilities, the handler func remains installed after a signal has been delivered.

If a caught signal occurs during certain system calls, causing the call to terminate prematurely, the call is automatically restarted. In particular this can occur during a read or write(2) on a slow device (such as a terminal; but not a file) and during a wait(2).

The value of signal is the previous (or initial) value of func for the particular signal.

After a fork(2) or vfork(2) the child inherits all signals. Execve(2) resets all caught signals to the default action; ignored signals remain ignored.

RETURN VALUE

The previous action is returned on a successful call. Otherwise, -1 is returned and *errno* is set to show the error.

ERRORS

Signal will fail and no action will take place if one of the following occur:

[EINVAL] Sig is not a valid signal number.

[EINVAL] An attempt is made to ignore or supply a handler for SIGKILL or SIGSTOP.

[EINVAL] An attempt is made to ignore SIGCONT (by default SIGCONT is ignored).

SEE ALSO

kill(1), ptrace(2), kill(2), sigvec(2), sigblock(2), sigsetmask(2), sigpause(2), sigstack(2), setjmp(3), tty(4)

NOTES

See "Notes" in sigvec(2) for information on declaring the signal handler routine.

```
NAME
```

sin, cos, tan, asin, acos, atan, atan2 - trigonometric functions and their inverses

SYNOPSIS

```
#include < math.h >
double sin(x)
double x;
double cos(x)
double x;
double tan(x)
double x;
double asin(x)
double acos(x)
double acos(x)
double x;
double atan(x)
double x;
double atan(x)
double y, x;
```

DESCRIPTION

Sin, cos and tan return trigonometric functions of radian arguments.

Asin returns the arc sin in the range $-\pi/2$ to $\pi/2$.

Acos returns the arc cosine in the range 0 to π .

At an returns the arc tangent of x in the range $-\pi/2$ to $\pi/2$.

Atan2 returns

```
atan(y/x) 	 if x > 0,
sign(y)*(\pi - atan(|y/x|)) 	 if x < 0,
x 	 if x = 0 	 and y = +0,
sign(y)*\pi 	 if x = 0 	 and y = -0, 	 and
sign(y)*\pi/2 	 if x = 0 	 but y 	 not 	 equal to 0.
```

DIAGNOSTICS

Arguments of magnitude greater than 1 cause asin and acos to return a NaN; errno is set to EDOM.

ERRORS (due to roundoff etc.)

Let P stand for the number stored in the computer in place of $\pi = 3.14159\ 26535\ 89793\ 23846\ 26433\ ...$ Let "trig" stand for one of "sin", "cos" or "tan". Then the expression "trig(x)" in a program actually produces an approximation to trig($x*\pi/P$), and "atrig(x)" approximates $(P/\pi)*$ atrig(x).

P differs from π by a fraction of an *ulp*; the difference matters only if the argument x is huge, and even then the difference is likely to be swamped by the uncertainty in x. Besides, every trigonometric identity that does not involve π explicitly is satisfied equally well regardless of whether $P = \pi$. For instance, sin(x) + cos(x) = 1 and sin(2x) = 2sin(x)cos(x) to within a few *ulps* no matter how big x may be. Therefore the difference between P and π is most unlikely to affect scientific and engineering computations.

SIN(3M)

NOTES

The three trig functions: sin, cos, and tan call static routines $cos_C()$ and $sin_S()$. When profiling programs calling the trig functions, use the -a flag for prof(1) or compile with -pg and use gprof(1). This will give you statistics for the static routines.

SEE ALSO

hypot(3M), math(3M), sqrt(3M)

AUTHORS

Robert P. Corbett, W. Kahan, Stuart McDonald, Kwok-Choi Ng

SINH(3M)

```
NAME
       sinh, cosh, tanh - hyperbolic functions
SYNOPSIS
       #include < math.h >
       double sinh(x)
       double x;
       double cosh(x)
       double x;
       double tanh(x)
       double x;
DESCRIPTION
       These functions compute the designated hyperbolic functions for real arguments.
DIAGNOSTICS
       Sinh and \cosh return \infty if the correct value would overflow.
ERRORS (due to roundoff, etc.)
       Below 2.4 ulps; an ulps is one Unit in the Last Place.
SEE ALSO
       math(3M)
AUTHORS
       W. Kahan, Kwok-Choi Ng
```

SQRT(3M)

NAME

cbrt, sqrt - cube root, square root

SYNOPSIS

#include < math.h >

double cbrt(x)

double x;

double sqrt(x)

double x;

DESCRIPTION

Cbrt returns the cube root of x.

Sqrt returns the square root of x.

DIAGNOSTICS

Sqrt returns a NaN when x is negative; errno is set to EDOM.

ERRORS (due to roundoff, etc.)

Cbrt is accurate to within 0.7 ulps.

Sqrt is correctly rounded in accordance with the rounding mode in force; the error is less than half an ulp in the default mode (round-to-nearest). An ulp is one Unit in the Last Place carried.

SEE ALSO

math(3M)

AUTHOR

W. Kahan

SYSLOG(3)

NAME

syslog, openlog, closelog, setlogmask - control system log

SYNOPSIS

```
#include < syslog.h >
```

openlog(ident, logopt, facility)

char *ident;

syslog(priority, message, parameters ...)

char *message;

closelog()

setlogmask(maskpri)

DESCRIPTION

Syslog arranges to write message onto the system log maintained by syslogd(8). The message is tagged with priority. The message looks like a printf(3) string except that %m is replaced by the current error message (collected from errno). A trailing newline is added if needed. This message will be read by syslogd(8) and written to the system console, log files, or forwarded to syslogd on another host as appropriate.

Priorities are encoded as a facility and a level. The facility describes the part of the system generating the message. The level is selected from an ordered list:

LOG_EMERG A panic condition. This is normally broadcast to all users.

LOG_ALERT A condition that should be corrected immediately, such as a corrupted sys-

tem database.

LOG_CRIT Critical conditions, e.g., hard device errors.

LOG_ERR Errors.

LOG WARNING Warning messages.

LOG_NOTICE Conditions that are not error conditions, but should possibly be handled

specially.

LOG_INFO Informational messages.

LOG_DEBUG Messages that contain information normally of use only when debugging a

program.

If syslog cannot pass the message to syslogd, it will attempt to write the message on |dev|console if the LOG_CONS option is set (see below).

If special processing is needed, openlog can be called to initialize the log file. The parameter ident is a string that is prepended to every message. Logopt is a bit field indicating logging options. Current values for logopt are:

LOG_PID log the process id with each message: useful for identifying instantiations of

daemons.

LOG_CONS Force writing messages to the console if unable to send it to syslogd. This

option is safe to use in daemon processes that have no controlling terminal

since syslog will fork before opening the console.

LOG_NDELAY Open the connection to syslogd immediately. Normally the open is delayed

until the first message is logged. Useful for programs that need to manage

the order in which file descriptors are allocated.

LOG_NOWAIT Don't wait for children forked to log messages on the console. This option

should be used by processes that enable notification of child termination

via SIGCHLD, as syslog may otherwise block waiting for a child whose

SYSLOG(3)

exit status has already been collected.

The facility parameter encodes a default facility to be assigned to all messages that do not have an explicit facility encoded:

LOG_KERN Messages generated by the kernel. These cannot be generated by any user

processes.

LOG_USER Messages generated by random user processes. This is the default facility

identifier if none is specified.

LOG MAIL The mail system.

LOG_DAEMON System daemons, such as flpd(8), routed(8), etc.

LOG_AUTH The authorization system: login(1), su(1), getty(8), etc.

LOG LPR The line printer spooling system: lpr(1), lpc(8), lpd(8), etc.

LOG RFS Remote file systems, currently the Andrew File System and RVD.

LOG_LOCAL0 Reserved for local use. Similarly for LOG_LOCAL1 through

LOG LOCAL7.

Closelog can be used to close the log file.

Setlogmask sets the log priority mask to maskpri and returns the previous mask. Calls to syslog with a priority not set in maskpri are rejected. The mask for an individual priority pri is calculated by the macro LOG_MASK(pri); the mask for all priorities up to and including toppri is given by the macro LOG_UPTO(toppri). The default allows all priorities to be logged.

EXAMPLES

```
syslog(LOG_ALERT, "who: internal error 23");
```

```
openlog("ftpd", LOG_PID, LOG_DAEMON);
setlogmask(LOG_UPTO(LOG_ERR));
syslog(LOG_INFO, "Connection from host %d", CallingHost);
```

syslog(LOG_INFO|LOG_LOCAL2, "foobar error: %m");

SEE ALSO

logger(1), syslogd(8)

INTRO(3F)

NAME

intro - introduction to FORTRAN library functions

DESCRIPTION

This section describes the functions in the FORTRAN run time library. These functions provide an interface from f77 programs to the system in the same manner as the C library does for C programs. They are automatically loaded as needed by the Fortran compiler f77(1).

Most of these functions are in libU77.a. Some are in libF77.a or libI77.a. A few intrinsic functions are described for the sake of completeness.

This release supports the 4.2/RT version of f77(1) and the run time library. It does not support the 4.3BSD level of the compiler. The following sections contain lists of the supported run time functions and of the 4.3BSD run time functions that are not supported.

For efficiency, the SCCS ID strings are not normally included in the a.out file. To include them, simply declare

external f77lid

in any f77 module.

LIST OF SUPPORTED FUNCTIONS

Name	Appears on P	age Description
abort	abort.3f	terminate abruptly with memory image
access	access.3f	determine accessability of a file
alarm	alarm.3f	execute a subroutine after a specified time
and	bit.3f	bitwise and
bessel	bessel.3f	of two kinds for integer orders
chdir	chdir.3f	change default directory
chmod	chmod.3f	change mode of a file
ctime	time.3f	return system time
dffrac	flmin.3f	return extreme values
dflmax	flmin.3f	return extreme values
dflmin	flmin.3f	return extreme values
drand	rand.3f	return random values
dtime	etime.3f	return elapsed execution time
etime	etime.3f	return elapsed execution time
exit	exit.3f	terminate process with status
fdate	fdate.3f	return date and time in an ASCII string
ffrac	flmin.3f	return extreme values
fgetc	getc.3f	get a character from a logical unit
flmax	flmin.3f	return extreme values
flmin	flmin.3f	return extreme values
flush	flush.3f	flush output to a logical unit
fork	fork.3f	create a copy of this process
fpecnt	trpfpe.3f	trap and repair floating point faults
fputc	putc.3f	write a character to a fortran logical unit
fseek	fseek.3f	reposition a file on a logical unit
fstat	stat.3f	get file status
ftell	fseek.3f	reposition a file on a logical unit
gerror	perror.3f	get system error messages
getarg	getarg.3f	return command line arguments
getc	getc.3f	get a character from a logical unit
getcwd	getcwd.3f	get pathname of current working directory
getenv	getenv.3f	get value of environment variables

INTRO(3F)

getgid	getuid.3f	get user or group ID of the caller
getlog	getlog.3f	get user's login name
getpid	getpid.3f	get process id
getuid	getuid.3f	get user or group ID of the caller
gmtime	time.3f	return system time
hostnm	hostnm.3f	get name of current host
iargc	getarg.3f	return command line arguments
idate	idate.3f	return date or time in numerical form
ierrno	perror.3f	get system error messages
index	index.3f	tell about character objects
inmax	flmin.3f	return extreme values
intro	intro.3f	introduction to FORTRAN library functions
ioinit	ioinit.3f	change f77 I/O initialization
irand	rand.3f	return random values
isatty	ttynam.3f	find name of a terminal port
itime	idate.3f	return date or time in numerical form
kill	kill.3f	send a signal to a process
len	index.3f	tell about character objects
link	link.3f	make a link to an existing file
lnblnk	index.3f	tell about character objects
loc	loc.3f	return the address of an object
long	long.3f	integer object conversion
lshift	bit.3f	left shift
lstat	stat.3f	get file status
ltime	time.3f	return system time
not	bit.3f	bitwise complement
or .	bit.3f	bitwise or
perror	perror.3f	get system error messages
putc	putc.3f	write a character to a fortran logical unit
qsort	qsort.3f	quick sort
rand	rand.3f	return random values
rename	rename.3f	rename a file
rindex	index.3f	tell about character objects
rshift	bit.3f	right shift
short	long.3f	integer object conversion
signal	signal.3f	change the action for a signal
sleep	sleep.3f	suspend execution for an interval
stat	stat.3f	get file status
system	system.3f	execute a UNIX command
tclose	topen.3f	f77 tape I/O
time	time.3f	return system time
topen	topen.3f	f77 tape I/O
traper	traper.3f	trap arithmetic errors
trapov	trapov.3f	trap and repair floating point overflow
tread	topen.3f	f77 tape I/O
trewin	topen.3f	f77 tape I/O
trpfpe	trpfpe.3f	trap and repair floating point faults
tskipf	topen.3f	f77 tape I/O
tstate	topen.3f	f77 tape I/O
ttynam	ttynam.3f	find name of a terminal port
twrite	topen.3f	f77 tape I/O
unlink	unlink.3f	remove a directory entry
wait	wait.3f	wait for a process to terminate
		K

INTRO(3F) INTRO(3F)

xor bit.3f bitwise exclusive or LIST OF UNSUPPORTED 4.3BSD FUNCTIONS Appears on Page Name Description plot.3f f77 interface to plot(3x) arc plot.3f f77 interface to plot(3x)box circle plot.3f f77 interface to plot(3x)plot.3f f77 interface to plot(3x) clospl plot.3f f77 interface to plot(3x)cont drandm random.3f better random number generator plot.3f f77 interface to plot(3x) erase falloc malloc.3f memory allocator memory allocator free malloc.3f better random number generator irandm random.3f

plot.3f

label

symlnk

plot.3f f77 interface to plot(3x) line linemd plot.3f f77 interface to plot(3x) malloc malloc.3f memory allocator f77 interface to plot(3x)move plot.3f openpl plot.3f f77 interface to plot(3x)point plot.3f f77 interface to plot(3x)random random.3f better random number generator space plot.3f f77 interface to plot(3x) symlnk.3f

f77 interface to plot(3x)

make a symbolic link

ABORT(3F) ABORT(3F)

NAME

abort - terminate abruptly with memory image

SYNOPSIS

subroutine abort (string)
character*(*) string

DESCRIPTION

Abort cleans up the I/O buffers and then aborts producing a core file in the current directory. If string is given, it is written to logical unit 0 preceded by "abort:".

FILES

/usr/lib/libF77.a

SEE ALSO

abort(3)

BUGS

String is ignored on the PDP11.

BESSEL(3F)

BESSEL(3F)

```
NAME
        Bessel functions - of two kinds for integer orders
SYNOPSIS
        function besj0 (x)
        function besj1 (x)
        function besjn (n, x)
        function besy0 (x)
        function besy1 (x)
        function besyn (n, x)
        double precision function dbesj0 (x)
        double precision x
        double precision function dbesj1 (x)
        double precision x
        double precision function dbesin (n, x)
        double precision x
        double precision function dbesy0 (x)
        double precision x
        double precision function dbesy1 (x)
        double precision x
        double precision function dbesyn (n, x)
        double precision x
DESCRIPTION
        These functions calculate Bessel functions of the first and second kinds for real arguments and
        integer orders.
DIAGNOSTICS
        Negative arguments cause besy0, besy1 and besyn to return -\infty. Errno is set to EDOM (33).
FILES
        /usr/lib/libF77.a
SEE ALSO
```

j0(3M), perror(3F)

PERROR(3F) PERROR(3F)

NAME

perror, gerror, ierrno – get system error messages

SYNOPSIS

subroutine perror (string) character*(*) string

subroutine gerror (string) character*(*) string

character*(*) function gerror()

function ierrno()

DESCRIPTION

Perror will write a message to fortran logical unit 0 appropriate to the last detected system error. String will be written preceding the standard error message.

Gerror returns the system error message in character variable string. Gerror may be called either as a subroutine or as a function.

Ierrno will return the error number of the last detected system error. This number is updated only when an error actually occurs. Most routines and I/O statements that might generate such errors return an error code after the call; that value is a more reliable indicator of what caused the error condition.

FILES

/usr/lib/libU77.a

SEE ALSO

intro(2), perror(3)

D. L. Wasley, Introduction to the f77 I/O Library

BUGS

String in the call to perror can be no longer than 127 characters.

The length of the string returned by gerror is determined by the calling program.

NOTES

UNIX system error codes are described in intro(2). The f77 I/O error codes and their meanings are:

- 100 "error in format" "illegal unit number" 101 "formatted io not allowed" 102 103 "unformatted io not allowed" 104 "direct io not allowed" 105 "sequential io not allowed" "can't backspace file" 106 "off beginning of record" 107 108 "can't stat file" 109 "no * after repeat count" "off end of record" 110 "truncation failed" 111 "incomprehensible list input" 112 113 "out of free space"
- 114 "unit not connected"
- "read unexpected character" 115
- "blank logical input field" 116

117	"'new' file exists"
118	"can't find 'old' file"
119	"unknown system error"
120	"requires seek ability"
121	"illegal argument"
122	"negative repeat count"
123	"illegal operation for unit"

RAND(3F)

NAME

rand, drand, irand - return random values

SYNOPSIS

function irand (iflag)

function rand (iflag)

double precision function drand (iflag)

DESCRIPTION

These functions use rand(3C) to generate sequences of random numbers. If iflag is "1", the generator is restarted and the first random value is returned. If iflag is otherwise non-zero, it is used as a new seed for the random number generator, and the first new random value is returned.

Irand returns positive integers in the range 0 through 2147483647. Rand and drand return values in the range 0. through 1.0.

FILES

/usr/lib/libF77.a

SEE ALSO

rand(3C)

TRAPER(3F) TRAPER(3F)

NAME

traper - trap arithmetic errors

SYNOPSIS

integer function traper (mask)

DESCRIPTION

NOTE: This routine is a null routine. IEEE default exception handling automatically provides an appropriate return value for exceptional operations. *Fpecount* always returns zero. What follows is the text of the original man page dated 18 July 1983.

NOTE: This routine applies only to the VAX. It is ignored on the PDP11.

Integer overflow and floating point underflow are not normally trapped during execution. This routine enables these traps by setting status bits in the process status word. These bits are reset on entry to a subprogram, and the previous state is restored on return. Therefore, this routine must be called *inside* each subprogram in which these conditions should be trapped. If the condition occurs and trapping is enabled, signal SIGFPE is sent to the process. (See signal(3C))

The argument has the following meaning:

value meaning

- 0 do not trap either condition
- 1 trap integer overflow only
- 2 trap floating underflow only
- 3 trap both the above

The previous value of these bits is returned.

FILES

/usr/lib/libF77.a

SEE ALSO

signal(3C), signal(3F)

TRAPOV(3F) TRAPOV(3F)

NAME

trapov - trap and repair floating point overflow

SYNOPSIS

subroutine trapov (numesg, rtnval) double precision rtnval

DESCRIPTION

NOTE: This routine is a null routine. IEEE default exception handling automatically provides an appropriate return value for exceptional operations. *Fpecount* always returns zero. What follows is the text of the original man page dated 18 July 1983.

NOTE: This routine applies only to the older VAX 11/780's. VAX computers made or upgraded since spring 1983 handle errors differently. See *trpfpe*(3F) for the newer error handler. This routine has always been ineffective on the VAX 11/750. It is a null routine on the PDP11.

This call sets up signal handlers to trap arithmetic exceptions and the use of illegal operands. Trapping arithmetic exceptions allows the user's program to proceed from instances of floating point overflow or divide by zero. The result of such operations will be an illegal floating point value. The subsequent use of the illegal operand will be trapped and the operand replaced by the specified value.

The first numes occurrences of a floating point arithmetic error will cause a message to be written to the standard error file. If the resulting value is used, the value given for rtnval will replace the illegal operand generated by the arithmetic error. Rtnval must be a double precision value. For example, "0d0" or "dflmax()".

FILES

/usr/lib/libF77.a

SEE ALSO

range(3F), signal(3F), trpfpe(3F)

BUGS

Other arithmetic exceptions can be trapped but not repaired.

There is no way to distinguish between an integer value of 32768 and the illegal floating point form. Therefore such an integer value may get replaced while repairing the use of an illegal operand.

TRPFPE(3F)

NAME

trpfpe, fpecnt - trap and repair floating point faults

SYNOPSIS

subroutine trpfpe (numesg, rtnval) double precision rtnval

integer function fpecnt ()

common /fpeflt/ fperr logical fperr

DESCRIPTION

NOTE: This routine is a null routine. IEEE default exception handling automatically provides an appropriate return value for exceptional operations. *Fpecount* always returns zero. What follows is the text of the original man page dated 26 July 1983.

NOTE: This routine applies only to VAX computers. It is a null routine on the PDP11.

Trpfpe sets up a signal handler to trap arithmetic exceptions. If the exception is due to a floating point arithmetic fault, the result of the operation is replaced with the *rtnval* specified. Rtnval must be a double precision value. For example, "0d0" or "dflmax()".

The first numesg occurrences of a floating point arithmetic error will cause a message to be written to the standard error file. Any exception that can't be repaired will result in the default action, typically an abort with core image.

Fpecnt returns the number of faults since the last call to trpfpe.

The logical value in the common block labelled fpeflt will be set to .true. each time a fault occurs.

FILES

/usr/lib/libF77.a

SEE ALSO

range(3F), signal(3F)

BUGS

This routine works only for faults, not traps. This is primarily due to the VAX architecture.

If the operation involves changing the stack pointer, it can't be repaired. This seldom should be a problem with the f77 compiler, but such an operation might be produced by the optimizer.

The POLY and EMOD opcodes are not dealt with.

Section 4. Special Files

This section describes special files, related driver functions, and networking support. Man pages found in IBM/4.3, but not in 4.3BSD, are marked with an asterisk (*).

- intro
- aedemul*
- ap*
- asy*
- autoconf
- bufemul*
- bus*
- cons
- disk*
- fd*
- hd*
- ibm5081*
- ibm5151*
- ibm5154*
- ibm6153*
- ibm6154*
- ibm6155*
- ibm8514*

- ibmaed*
- ibmemul*
- kbdemul*
- lan*
- lp
- mem
- mouse*
- mtio
- psp*
- rvd(4P)*
- sc*
- speaker*
- st*
- stdemul*
- tb
- tty
- un*
- vga*
- xemul*

INTRO(4)

NAME

intro – introduction to special files and hardware support

DESCRIPTION

Intro(4) describes the special files, related driver functions, and networking support available in the system. In this part of the manual, the "Synopsis" section of each configurable device gives a sample specification for use in constructing a system description for the config(8) program. The "Diagnostics" section lists messages that may appear on the console and in the system-error log as a result of errors in device operation.

This section contains devices that may be configured into the system -- "4" entries -- as well as network-related information ("4N", "4P" and "4F" entries). The networking support is introduced in *intro*(4N).

IBM RT PC and IBM 6152 Academic System DEVICE SUPPORT

This section describes the hardware supported on the IBM RT PC and on the IBM 6152 Academic System. Software support for the devices comes in two forms. A hardware device can be supported with a character or block device driver, or it can be used within the networking subsystem and have a network interface driver. Block and character devices are accessed through files in the file system of a special type; compare with mknod(8). Network interfaces are indirectly accessed through the interprocess communication facilities provided by the system (see socket(2)).

A hardware device is identified to the system at configuration time and the appropriate device or network interface driver is then compiled into the system. When the resultant system is booted, the autoconfiguration facilities in the system probe for the device and, if found, enable the software support for it. If a device does not respond at autoconfiguration time, it is not accessible at any time thereafter. To enable a device that did not autoconfigure, reboot the system.

The autoconfiguration system is described in *autoconf*(4). A list of the supported devices is given below.

DEVICES

asy

The devices listed below are supported under PRPQ #5799-CGZ for the IBM RT PC; they are indicated by their functional interfaces.

Multi-port asynchronous communications RS232C interface

bus	Control of access to the system I/O bus	
fd	Floppy disk drive interface	
hd	PC/AT, ESDI, and EESDI hard disk interface	
ibm5081	IBM 5081 Display interface (mpel)	
ibm5151	IBM 5151 Monochrome Display interface (mono)	
ibm5154	IBM 5154 Enhanced Graphics Display interface (ega)	
ibm6153	IBM 6153 Advanced Monochrome Graphics Display interface (apa8)	
ibm6154	IBM 6154 Advanced Color Graphics Display interface (apa8c)	
ibm6155	IBM 6155 Extended Monochrome Graphics Display interface (apa16)	
ibmaed	IBM Academic Information Systems experimental display interface (aed)	
lan	IBM RT PC Token-Ring Adapter	
lp	Line printer interface	
mouse	Mouse interface	
psp	Planar serial port RS232 interface	
sc	IBM 9332 disk unit using the IBM Small Computer System Interface (SCSI) Adapter	
speaker	Console speaker interface	
st	Streaming tape interface	
un	IBM RT PC Baseband Adapter for use with Ethernet	

INTRO(4)

The devices listed below are supported for the IBM 6152 Academic System; they are indicated by their functional interfaces.

asy Asynchronous communications RS232C interface

fd Floppy disk drive interface hd PC/AT hard disk interface ibm8514 IBM 8514/A Display interface

lp Line printer interface mouse Mouse interface

lan IBM Token-Ring Network PC Adapter un Ungermann-Bass 10Mb/s Ethernet interface

vga PS/2 Display interface

The following emulators provide a unified way for several different (but related) functions to communicate with the operating system kernel. These functions provide operator input, display output, and console buffering.

aedemul Graphics interfaces for IBM Academic Information Systems experimental display

(for the IBM RT PC only)

bufemul Kernel buffering emulator

ibmemul IBM 3101 emulator

kbdemul Default keyboard emulator

stdemul Standard output emulator (for the IBM RT PC only)
xemul X input emulator for queuing keyboard and mouse events

SEE ALSO

autoconf(4), intro(4N), config(8)

AEDEMUL(4) AEDEMUL(4)

NAME

aedemul - graphics interfaces for the IBM Academic Information Systems experimental display

SYNOPSIS

pseudo-device aed

DESCRIPTION

Aedemul supplies a graphics interface used by aedlib (see intro(3)). Direct user access can be obtained by the following code:

Once the above is done, the following system calls are supported by /dev/aed:

close Terminates a process's read/write access to the display's shared RAM and reinstates glass tty mode.

read Allows read access to the display's control store.

write Allows write access to the display's control store.

ioctl (Ioctl's defined in < sys/aedioctl.h>) AEDSEM_WHILE, AEDSEM_UNTIL, AEDSEM_SET_WAIT all poll the semaphore, waiting for an event. A specified number of polls occurs before the call returns with a failing (-1) return code. You may alter the default of 250 polls by issuing a AEDSEM_TIMEOUT request. It is often advisable to reissue the failed call later, allowing the display a little more time. It is up to the programmer to decide how many tries make sense before the display can be assumed to be hung. The following requests are supported:

IOCINFO Returns device information.

IOCTYPE Returns device type.

AEDGET_SRAM LOC

Reports address of display-shared RAM.

AEDRESET Resets the display. The display processor is halted and the program counter is set to 0.

AEDSEM_READ Reads the display semaphore.

AEDEMUL(4) AEDEMUL(4)

AEDSEM SET Sets display semaphore to a specified value.

AEDSEM_SET_WAIT Sets display semaphore to a specified value and waits for it to

change, then returns the new value.

AEDSEM WHILE Polls display semaphore until it is no longer equal to a specified

value.

AEDSEM_UNTIL Polls display semaphore until it is equal to a specified value.

AEDSEM_TIMEOUT Sets the timeout interval. The default is 250 polls. See

AEDSEM_WIIILE, AEDSEM_UNTIL, AEDSEM_SET_WAIT.

A request to set the timeout to zero returns the current

timeout value, leaving it unaffected.

AEDSTART Resumes display microprogram execution.

AED_BEEP Causes an audible beep.

AED LEDS Writes a value to the system-unit LEDs. The number written must be a

hex number with both digits less than A. For example, "0x34" displays "34", but "0x4F" behaves unpredictably. Remember, the system updates the LEDs

frequently, so if you want to see your numbers, write them often.

AEDDELAY Does an I/O delay, using the planar I/O delay register.

AEDSTATE Returns an integer indicating the state of the display.

AEDSTOP Stops display microprogram execution.

FILES

/dev/console /dev/aed

SEE ALSO

aedjournal(1), aedrunner(1), intro(3G), bus(4), ibmaed(4), tty(4), aedtest(8) "IBM/4.3 Console Emulators", in Volume II, Supplementary Documents

BUGS

The emulator is valid only for the IBM Academic Information Systems experimental display, not for the other supported apa devices.

NAME

ap - asynchronous data mode protocol line discipline

SYNOPSIS

pseudo-device ap

DESCRIPTION

The ap line discipline is used to communicate with the IBM 3812 Pageprinter through a serial adapter in asynchronous data mode. The ap line discipline detects and recovers from line errors, establishes an expedited command path to the printer logically independent from the data path, and returns printer status information to the user. The normal system calls open(2), close(2), write(2), read(2), and ioctl(2) are used. There can be only one program at a time using ap on a given serial port. The program may consist of more than one process.

To use the ap line discipline, open the serial port, use ioctl to set the appropriate speed (typically 19.2 kilobaud) with no parity (set both EVENP and ODDP), and issue the TIOCSETD ioctl with the argument APLDISC. After ap has control, the only ioctl commands that are valid are APRDCMD, APTEST, APTESTRD, APWRTCMD, TIOCGETD, TIOCSETD, TIOCGETP, TIOCEXCL, and TIOCNXCL.

Data is written to ap using write. Expedited commands are written using ioctl. Only one write operation can be done at a time, whether by write or by ioctl. Data is read using read. Expedited commands are read using ioctl. If there is pending data or an expedited command to be read, a write may write only part of the data. The number of bytes actually written will be returned. If the read or write returns a -1, and errno is set to EIO, then either the line is down, or there is data or an expedited command to be read from the 3812. Ioctl is used to determine the condition that terminated the write or caused the EIO.

The *ioctl* calls that apply to *ap* have the form:

```
#include < machineio/apio.h >
    #include < sys/tty.h >
    ioctl(fildes,code,arg)
    struct apioinfo *arg;
where:
    struct apioinfo {
        int apstatus; /* status code */
        char apcmd; /* expedited command */
};
```

The *ioctl* codes are:

APRDCMD

Read an expedited command from the 3812. This command will block waiting for an expedited command. It will return an EIO if the connection is down, or if there is data to be read from the 3812.

APTEST

Return the status of ap. It will not read an expedited command. If an expedited command is available to be read, the status field in arg is set to APGOTCMD.

APTESTRD

Return the status of ap and return expedited commands. If an expedited command is available to be read, it is read into arg. If there is nothing to read, and no status to report, the status field in arg is set to APNOINFO.

APWRTCMD

Send an expedited command to the 3812. Arg points to the expedited command.

The status codes are:

APDOWN

The 3812 has lost synchronization. The file descriptor for the serial port must be closed and the ap line discipline restarted.

APGOTDATA

Data has been received from the 3812 and is available to be read from ap.

APGOTCMD

An expedited command byte has been received from the 3812. It is stored in arg if the ioctl was APRDCMD or APTESTRD.

APNOINFO

There is no data nor expedited command to be read, and the line is not down. It is returned only for APTEST and APTESTRD.

APSYNCING

Synchronization with the 3812 is in process but is not complete.

NOTE

Expedited commands are described in the IBM 3812 Pageprinter Programming Reference, S544-3268.

SEE ALSO

tty(4)

IBM 3812 Pageprinter Programming Reference, S544-3268

ERRORS

[EIO] A read, write, or iocil encountered a condition that prevented its execution. Issue the APTEST or APTESTRD iocil to determine the condition.

[ENOMEM]

This condition is encountered when the TIOCSETD *ioctl* is executed to switch to APLDISC, and the line discipline can not acquire the necessary buffer space.

Incomplete write

A write that returns a non-negative count less than the count requested indicates an outstanding condition that should be checked with the APTEST or APTESTRD ioctl. The returned count indicates the number of bytes successfully written. After clearing the condition, adjust the parameters and continue the write.

NAME

asy - multi-port asynchronous communications RS232C interface

SYNOPSIS

For the IBM RT PC:

device asy0 at iocc0 csr 0xf0001230 flags 0x01 device asy1 at iocc0 csr 0xf0002230 flags 0x0f device asy2 at iocc0 csr 0xf0003230 flags 0x0f device asy3 at iocc0 csr 0xf0004230 flags 0x0f device asy4 at iocc0 csr 0xf00003f8 flags 0x00 device asy5 at iocc0 csr 0xf00002f8 flags 0x00

For the IBM 6152 Academic System:

device asy4 at iocc csr 0x000003f8 flags 0xff priority 4

DESCRIPTION

The IBM RT PC multi-port asynchronous communications adapter card provides four serial RS232C ports, asy0 through asy3.

The IBM RT PC can support up to four adapters. Four separate I/O address ranges (0xf0001230, 0xf0002230, 0xf0003230 and 0xf0004230) and three separate interrupt levels (IRQ9, IRQ10, IRQ11) are reserved for the multiport adapters. Two adapters may share the same interrupt level, but not the same I/O address range.

On the IBM RT PC, this driver also supports the serial line of up to two PC/AT serial/parallel adapters. These adapters must be specified as asy4 and/or asy5 in the *config*(8) file even if you do not have any multi-port adapters configured.

On the IBM 6152 Academic System, this driver supports the planar serial port on the PS/2. This must be specified as asy4 in the config(8) file.

The adapter supports only asynchronous communications, and is fully programmable. Each port behaves as described in tty(4) and may be set to run at any of 16 speeds; see tty(4) for the encoding.

Bit i of flags may be set to 1 to specify that the ith port should be treated as having carrier initially present. For example, specifying "flags 0x01" in the configuration entry for asy0 causes only the line tty00 to be treated in this way.

FILES

For the IBM RT PC:

/dev/tty{00,01,...,15} for multi-port cards /dev/ttyc[01] for PC/AT serial/parallel adapters /dev/ttyd[0-9a-f] (dialups)

For the IBM 6152 Academic System: /dev/ttyc0 for PS/2 planar serial port

SEE ALSO

tty(4)

DIAGNOSTICS

asy%d: overrun error. Data has been lost because characters are arriving at the adapter faster than the driver can read them.

NOTES

This driver should also support the IBM RT PC RS422A protocol in so far as the programming interface is the same. However, this has not been tested.

This page intentionally left blank.

AUTOCONF(4) AUTOCONF(4)

NAME

autoconf - diagnostics from the autoconfiguration code

DESCRIPTION

When IBM/4.3 loads, it checks the machine on which it runs to locate controllers, drives and other devices, printing its findings on the console. This procedure is driven by a system-configuration table processed by *config*(8) and compiled into each kernel.

Planar and PC/AT devices on the IBM RT PC are located by checking to see whether their adapter registers respond. If not, the devices are silently ignored. If the control-status register responds, but the device cannot be made to interrupt, a diagnostic warning prints on the console and the device becomes unavailable to the system.

You can build a generic system that picks its root device at boot time as the "best" available device (e.g. hard disks are better than diskettes); the device must be drive 0 to be considered. If such a system is booted with the RB_ASKNAME option on (see reboot(2)), the name of the root device is read from the console terminal at boot time, and any available device may be used.

SEE ALSO

intro(4), config(8)

DIAGNOSTICS

%s%d adapter %x IRQ %d CPU level %d. The device %s%d, e.g. asy0, was found "on" at adapter address %x (e.g. 0xf0001230), at PC/AT IRQ number %d (this is specified as the "priority" on the *config*(8) configuration) and interrupted at CPU level %d. The CPU level will be omitted if the the driver does not attempt to cause an interrupt.

%s%d at%s%d slave %d. A disk, diskette or tape drive that attaches to the indicated adapter was found. For disks, the message will be of the format "hd0 at hdc0 slave 0"; for diskettes, the format is "fd0 at fdc0 slave 0"; and for streaming tapes, "st0 at stc0 slave 0". The slave number pertains to the way in which the drive is jumpered or cabled. The device will be known as, for example, "hd0".

%s%d adapter %x didn't interrupt. The device did not interrupt, probably because it was broken, hung or not the device it was advertised to be.

8259 (%x) IRQ %d: no slih/unclaimed Stray Interrupt! level = %d, info = %x. An interrupt was not serviced by any driver.

autoconf: help-interrupt at irq %d and irq %d!. When a device probe routine was attempting to cause an interrupt, interrupts were received at two different IRQs. This can be caused by incorrect hardware switch or jumper settings.

AUTOCONF. Autoconfiguration is starting.

Null interrupt service routine for IRQ %d. No interrupt service routine was specified in the driver.

Warning: slow memory cards are not supported on APC. Machine checks are likely. Older memory cards cannot be used with the Advanced Processor Card. (This can also be caused by a missing or incorrect jumper on a fast memory card.) (IBM RT PC only)

Warning: 4MB of 8MB memory card unusable - 4MB card recommended. When two 8MB fast memory cards are used on an APC with 4MB of on-card memory, there are 4MB of memory that aren't usable. The first 8MB card can be replaced by a 4MB card. (IBM RT PC only)

Warning: 4MB of 8MB memory card unusable — exchange card order. Putting an 8MB fast memory card into the first memory slot wastes 4MB of memory. The 8MB card should be exchanged with whatever is in the second memory slot. (IBM RT PC only)

IOIM EC level different than expected. This indicates a hardware level different than the software expected.

AUTOCONF(4) AUTOCONF(4)

Warning: back-level IOIM - no reply register. The IOIM reply register does not appear to be functioning.

BUFEMUL(4)

BUFEMUL(4)

NAME

bufemul - kernel buffering emulator

DESCRIPTION

Bufemul is used to handle standard tty-type output to a display currently being used for graphics. Bufemul puts this output into a circular buffer which is sent to the display after it has been closed. Opening |dev|aed, |dev|apa16, |dev|apa8c, |dev|apa8, |dev|ega, |dev|mono, |dev|vga, |dev|ibm8514, or |dev|mpel automatically selects this emulator.

Bufemul supplies the following locals define in < machinecons/bufemul.h>:

BUFSETWIND Set the 128K window on the IBM 6152 Academic System. (See the IBM

6152 Academic System Technical Reference.) Arg points to a 24-bit PC memory address. Only addresses used by the VGA display are valid.

(IBM 6152 Academic System only.)

BUFINITVGA Initialize the VGA to one of its modes. Arg is the desired mode. See

vga(4) and < machinecons/vgaio.h > . (IBM 6152 Academic System only.)

BUFINIT8514 Initialize the 8514 to advanced function mode. No arguments are passed.

(IBM 6152 Academic System only.)

BUFDISPINFO Arg returns the following information about the display:

BUF_IS_ATR(arg) True when the CPU is an IBM 6152 Academic System.

BUF IS RTPC(arg) True when the CPU is an IBM RT PC.

BUF_GET_VGA(arg)

Get the type of display connected to the VGA.

BUF_GET_8514(arg)

Get the type of display connected to the IBM 8514/A. 0 = none, 1 = color, 2 = gray. Valid only the the IBM 6152 Academic System with the IBM 8514/A installed.

BUF GET EGA(arg)

Return the value of the switches on the EGA display. Valid only for the IBM RT PC with an EGA card installed.

FILES

For the IBM RT PC:

/dev/aed

/dev/apa16

/dev/apa8c

/dev/apa8

/dev/ega

/dev/mono

/dev/mpel

For the IBM 6152 Academic System:

/dev/vga

/dev/ibm8514

SEE ALSO

cons(4), ibmemul(4), kbdemul(4), stdemul(4), xemul(4)

"IBM/4.3 Console Emulators", in Volume II, Supplementary Documents

This page intentionally left blank.

NAME

bus - control of access to the system I/O bus

DESCRIPTION

Bus is a special file that controls a process's access to the I/O space (segment 0xf) of the IBM RT PC or the PCIF map (segment 0xd) on the IBM 6152 Academic System. Its use is usually associated with opening one of the bit-mapped display devices, when the user wants to read from or write to device control registers or display memory.

Unlike /dev/kmem (mem(4)), which restricts memory accesses to reads and writes through a file descriptor, opening /dev/bus allows the process direct access to the 64KB of I/O addresses and the 16MB of memory addresses on the system I/O bus. Closing /dev/bus removes these access rights.

No other operations on /dev/bus are provided.

NOTES

On the IBM 6152 Academic System, the I/O addresses are limited to those addresses used by display devices.

FILES

/dev/bus

SEE ALSO

ibm5081(4), ibm5151(4), ibm6153(4), ibm6154(4), ibm6155(4), ibm8514(4), ibmaed(4), mem(4), vga(4)

BUGS

On the IBM RT PC, opening /dev/bus allows access to all devices on the system I/O bus.

This page intentionally left blank.

CONS(4) CONS(4)

NAME

cons - keyboard and console display interface

DESCRIPTION

The keyboard and various possible displays combine to provide a terminal-like interface to the system. Internally, these are separate devices which software combines to emulate a normal terminal. See the appropriate manual pages for information about each display and the keyboard.

The keyboard adapter also supports the speaker, which is activated when the ASCII character bel (^G) is sent to the display with software. For additional information on speaker control, see speaker(4).

Console Device Control

The display devices, |dev|ttyaed, |dev|ttyap16, |dev|ttyap8c, |dev|ttyapa8, |dev|ttyega, |dev|ttymono, |dev|ttympel, |dev|ttyvga, and |dev|tty85/4 are all minor devices under |dev|console, and are all capable of displaying console output. Unique to this system is the fact that you may have one or more of these displays on your workstation at a time and any one can act as a console. With only one keyboard and system mouse, the console driver multiplexes these input devices to the many displays. All of the displays may have simultaneous logins and the user can "hot key" between each display. At first, this "input focus" is on the first device in the above sequence to be found at initialization time. The input focus can be manually switched to the next available display by pressing the default "hot key" < Alt > < Scroll Lock > . When the input focus is on a display, all keyboard and mouse data are sent to the process(es) that read from that display.

If no other console tty device is open, and only the default input emulator is used (see kbdemul(4)), the input focus is set to |dev|console. In this case, < Alt > < Scroll Lock > only switches which display gets console output. In the case where one or more tty devices are open, or the default input emulator changes, |dev|console gets no input. It tries to send output to the currently focused device. A user can redirect these console messages to any tty devices with the TIOCCONS joctl.

To support the many displays and the multiplexing between them, an emulator package was developed to work with the console driver. This package allows different types of emulation on input and output to be written independently of device.

The display devices |dev|aed, |dev|apa16, |dev|apa8c, |dev|apa8, |dev|ega, |dev|mono, |dev|mpel, |dev|vga, and |dev|ibm8514 are also minor devices to |dev|console. They are typically used by window managers and other graphic applications. When the focus is pointed to one of these display devices, the console messages are put in a circular buffer (see bufemul(4)) unless redirected with the TIOCCONS ioctl. The buffer is flushed to the screen upon closing the display device.

Controlled Access to Multiple Console Displays

It can be useful to deny access temporarily to one or more displays. For example, if the monochrome/printer adapter is present, the kernel assumes the presence of the monochrome display -- even if disconnected -- and attempts to initialize it. This can be blocked by modifying the access control bits for each display device (see < machinecons/consio.h>):

```
CONSDEV_KERNEL (this device available to kernel) CONSDEV_USER (this device available to user)
```

An ioctl, SCRIOCSETC (screen ioctl: set configuration), sets the desired access rights of a particular device. SCRIOCGETF (screen ioctl: get flags) gets the present state of a particular device:

```
CONSDEV_PRESENT(kernel has detected adapter)
CONSDEV_KERNEL
CONSDEV_USER
CONSDEV_INIT (device is initialized for console use)
CONSDEV_TTY (console tty device is open)
CONSDEV GRA (console graphic device is open)
```

CONS(4)

These ioctls are used by setscreen(8).

NOTE

On the IBM RT PC, the kernel flashes "98" on the LEDs if it cannot find any configured display during initialization, and then proceeds.

DIAGNOSTICS

None.

FILES

For the IBM RT PC:

/dev/console

/dev/aed

/dev/apa16

/dev/apa8c

/dev/apa8

/dev/ega

/dev/mono

/dev/mpel

For the IBM 6152 Academic System:

/dev/vga

/dev/ibm8514

SEE ALSO

bufemul(4), bus(4), ibm5081(4), ibm5151(4), ibm6153(4), ibm6154(4), ibm6155(4), ibm8514(4), ibmaed(4), ibmemul(4), kbdemul(4), speaker(4), stdemul(4), tty(4), vga(4), xemul(4), setscreen(8) "IBM/4.3 Console Emulators", in Volume II, Supplementary Documents

DISK(4)

NAME

disk - format of reserved areas of the hard disk

SYNOPSIS

```
#include < machineio/hdconfig.h > #include < machine/dkio.h >
```

DESCRIPTION

On the IBM 6152 Academic System, the first sector of all fixed disks must have a master fixed disk boot record. The first sector also contains the boot record partition table. This partition table defines how the disk is divided among up to four partitions. We will refer to each of these partitions as physical partitions, because they are contiguous disk space, aligned on clyinder boundries and are dedicated to a specific operating system.

Logical blocks are numbered beginning at zero and continuing across track and cylinder boundaries to the last block on the disk. Sectors on each track are numbered beginning at 1. The notation "(cylinder,track,sector)" denotes the physical location of a block. Thus, block 0 corresponds to sector (0,0,1).

Cylinder zero of the "c" partition is reserved by IBM for information describing the disk itself, including its geometry and the location of bad sectors. Since building a file system at cylinder zero would destroy this information, the default "a" partition begins at cylinder 1. The "c" partition may be used to access this information, but it is inadvisable to build a file system on a "c" partition. On the IBM RT PC, a portion of cylinder zero, beginning at (0,2,9), is reserved for use by hardware test routines during service procedures.

The last few cylinders of the "c" partition are also reserved; on the IBM RT PC, hardware test routines use the last one, and the preceding few cylinders provide the replacement pool for up to 1000 bad sectors. On the IBM 6152 Academic System, these cylinders provide replacement for up to 100 load sectors.

In the following structure definitions, Block n refers to the *n*th block of the "c" partition. It should be noted that not all of the following structures make sense in the IBM 6152 Academic System environment.

The following structures are included from < machine io/hdconfig.h>.

Block 0

Block 0 is the hard disk IPL record. It is described by the following structure:

```
struct boothdr {
         char
                  boot ibma[4];
                                             /* 0: IBMA in EBCDIC */
                                             /* 4: reserved */
                  boot check;
        long
                  boot lastcyl;
                                             /* 8: last available cyl number */
         short
                                             /* 10: last track number */
         char
                  boot lasttrack;
         char
                  boot lastsect;
                                             /* 11: last sector number */
                  boot sectorsize;
                                             /* 12-13: block/sector size (bytes) */
         short
                  boot reserved [4];
                                             /* 14-18: reserved */
         char
                                             /* 19: interleave factor */
         char
                  boot interleave;
                                             /* 20-23: reserved */
         char
                  boot reserved2[3];
        int
                  boot sectorcount;
                                             /* 24-27: disk size */
                  boot formatdate;
                                             /* 28-31: reserved */
        long
                  boot cyl;
                                             /* 32-33: cyl number of boot */
        short
                                             /* 34: track number of boot */
        char
                  boot track;
                                             /* 35: sector to boot */
        char
                  boot sector;
        long
                  boot length;
                                             /* 36-39: length in sectors */
        long
                  boot entry;
                                             /* 40-43: entry point */
                                             /* 44-47: block # of vrm minidisk */
                  boot vrmminidisk;
        long
```

DISK(4)

```
long boot_llp; /* 48-51: Loadlist processor block # */
long boot_vrmlength; /* 52-55: length of VRM minidisk */
char boot_fill[512-56]; /* reserved */
};
```

Block 1

Block 1 is the manufacturer's configuration sector. It is described by the following structure:

```
struct hdconfig {
                                     /** manufacturers configuration record */
         int
                  conf magic;
                                             /* 0: always 0xF8E9DACB */
                                             /* 4: number of sectors on device */
         int
                  conf sectorcount;
                  conf landing;
                                             /* 8: reserved */
         short
                  conf interleave;
                                             /* 10: interleave */
         char
                  conf sectsize;
                                             /* 11: sectors size 02 = 512 */
         char
         short
                  conf lastcyl;
                                             /* 12: last data cylinder */
                                             /* 14: last head number */
         char
                  conf lasttrack;
         char
                  conf lastsect;
                                             /* 15: last sector number */
         char
                  conf_precomp;
                                             /* 16: precomp or 0xff */
         char
                  conf status;
                                             /* 17: used by loadable post */
         short
                  conf maxcyl;
                                             /* 18: last physical cyl */
#define conf ce cyl conf maxcyl
         short
                  conf end of life;
                                             /* 20: max number of defects */
         struct seek curve {
                  short seek x;
                                             /* x position (cylinders?) */
                                             /* y position (milliseconds?) */
                  short seek y;
                                             /* slope millseconds/cylinder?) */
                  short seek slope;
         } conf seek[5];
                                             /* 22-51: seek curve characteristics */
         struct file id {
                  char
                                             /* to extend file size?? */
                           file fill1;
                  char
                           file size;
                                             /* approximate # megabytes */
                                             /* mfr code? */
                  char
                           file mfr;
         } conf file id;
                                             /* 52-54: manufacturer drive type id */
#define conf size conf file id file size
#define conf mfr conf file id.file mfr
         char
                  conf adapter;
                                             /* 55: adapter type */
#define HD ADAPTER AT
                                    0x00
#define HD_ADAPTER_ESDI
                                    0x01
#define HD ADAPTER HESDI 0x02
                                             /* 56-57: service request num */
         short
                  conf srn;
                                             /* 58: ascii drive type id */
         char
                  conf label;
         char
                  conf fill1[256-59];
                                             /* fill up to halfway */
         char
                  conf name[8];
                                             /* 256-263: name of disk (ACIS) */
         char
                  conf fill2[512-264];
                                             /* fill up to the end */
                                             /* 512: total size */
};
```

Block 8

Block 8 (0,0,9) is the beginning of the bad-block table. The first 6 bytes contain the word "DEFECT" in ASCII. This is followed by a short integer containing the number of entries in the table. Each entry consists of two longs; the first is the logical block number for the bad block and the second is the logical block number for the replacement block. The following structure may be used to map the bad-block table:

DISK(4)

```
struct hdbad {
                                            /* "DEFECT" */
        char
                 hddefect[6];
        short
                 hdcount;
                                            /* count of entries in table */
         struct hdmap {
                 unsigned hdreason:8,
                                            /* the reason block was unusable */
                          hdbad:24;
                                            /* the bad block number */
                                            /* the good block number */
                 int
                          hdgood:
                                            /* the table of bad to good blocks */
        } hdmap[];
};
Blocks 3-5, (0, 1, 15-17)
```

These blocks contain the primary and secondary copies of the VRM minidisk table which can be used to define non-standard 4.3/RT disk partitions. Any minidisk that has a name of the form "hdxy" where "y" is a letter between "a" and "h", will be taken as 4.3/RT partition "y".

In addition, a VRM paging minidisk is always taken as a "b" partition. On the IBM RT PC, if the minidisk table is not valid or if no valid 4.3/RT partitions are found, then the default partitions as given in *disktab*(5) are used. Because of variable size physical partitions on the IBM 6152 Academic System, minidisk information must be intialized and configured before a disk can be used for IBM/4.3. The minidisk directory has the following format:

```
* the following defines and structures relate to the VRM
 * minidisk directory
 */
#define ENDLIST -1
                           /* end of list marker */
#define START BLOCK(sectors) ((sectors)*4)
                                                      /* start of allocatable space */
struct miniheader {
        long number;
                                    /* number of entries */
         long level;
                                    /* revision level */
                                    /* index of next unused entry */
         short unused;
         short lastused;
                                    /* index of entry most recently used */
         short first:
                                    /* first entry */
         short last:
                                    /* last entry index */
        long bad block;
                                    /* first bad block */
        long bad size;
                                    /* number of bad blocks */
        long unused2;
                                    /* reserved */
        long unused3;
                                    /* reserved */
};
struct minidisk {
         short previous;
                                    /* 0 link to previous entry */
                                    /* 2 line to next entry */
         short next:
                                    /* 4 name of minidisk */
         char name[4];
         unsigned:32;
                                    /* 8 */
                                    /* 12 the date of creation */
        long date:
         short iodn;
                                    /* 16 the IODN used */
#define nextfree iodn
                                             hokey way of doing it! */
         char blocksize;
                                    /* 18 filesystem block size (bits 0-3) */
#define BLOCK 512
                           0
#define BLOCK 1024
                           1
#define BLOCK 2048
                           3
```

```
/* 19 type flags */
        char type;
#define TYPE WRITEVERIFY0x80
                                         /* verify writes */
#define TYPE_NOBADBLOCK
                                          /* no bad block forwarding on this disk */
                                 0x40
#define TYPE PAGE
                         0x20
                                 /* pageing space */
#define TYPE FILE
                                 /* AIX file system */
                         0x10
#define TYPE AIX
                                 /* AIX system minidisk */
                         0x08
#define TYPE PC
                                         /* coprocessor disk */
                                 0x04
#define TYPE VRM
                         0x02
                                 /* VRM */
#define TYPE IPL
                         0x01
                                 /* IPL'able */
        long unused;
                                 /* 20 */
        long start;
                                 /* 24 start block of partition */
        long size;
                                 /* 28 size of partition */
                                 /* 32 total length */
#define ISFREE(disk) ((disk)->iodn = = 0)
#ifdef KERNEL
#define MAXDISKS 23
                                 /* keep size down for kernel use */
#else
#define MAXDISKS 47
                                 /* actual minidisk size */
#endif KERNEL
                                         /* starting block number of minidisk */
#define MINIDISK BLOCK
                                 3
#define MINIDISK BLOCK2(sectors)
                                         ((sectors) + 14)
                                                          /* starting block number of minidisk copy
#define BLOCK SIZE
                         512
#define ROUND(a,b) (a+b-1)/b*b/* round up to next unit of b */
struct minidirectory {
        struct miniheader header:
        struct minidisk minidisk[MAXDISKS];
};
```

(0,1,18-21) (IBM RT PC EESDI only)

The hidden defect table starts with sector (0,1,18). This table exists only on hd70e (labelled "E70") disks. Such disks have been formatted so that only 35 of their 36 sectors on each track are available for general use. The 36th sector is used as a replacement block for the last bad block (if any) on the track. This organization means that bad blocks no longer cause time consuming seeks over to the high cylinders of the disk and back again. (Note that more than 1 bad block on a track is a very rare event.)

The first 6 bytes of the table contain the word "HIDDEN" in ASCII. This is followed by a short integer containing the number of entries in the table. Each entry consists of the physical cylinder, track, and sector of the hidden bad block. The implied replacement block is the last (36th) sector of the same track as the bad block. The following structure may be used to map the hidden-defect table:

```
} hidmap[510];
#define IGNORE HID 0xffffffff /* deleted hidmap entry */
The hd(4), sc(4), fd(4), and vdfR(4) disk drivers support the DKIOCGPART ioctl(2) which
returns the partition size and other optional information. The following structures are included from
< machineio/dkio.h > :
 * Structures and definitions for disk io control commands
/* disk io partition (and geometry) information format */
 * Notes:
 * 1.dk size may be less than dk cyl * dk track * dk sector
        since dk size is the usable size (e.g. what you would specify
        to newfs/mkfs)
 * 2.everything except dk size and dk blocksize are optional - if the
        information isn't known then zero is returned.
 * 3.name should be in domain of types in /etc/disktab.
 * 4.start is starting block of this partition (0 = start of disk)
        this could be used by a program to display current disk
        partitioning in a portable fashion.
                                            /* size of name field */
#define DK MAXNAME 16
struct dkpart {
                 dk size; /* disk partition size in blocksize units */
        u long
                                   /* see following word */
                                   /* block/sector size (in bytes) */
                 dk blocksize;
        u long
                 dk start; |* starting block (in blocksize units) *|
        u long
        u long
                dk ncyl; |* number of cylinders (if known) *|
        u long
                dk ntrack;
                                   /* number of tracks per cylinder (if known) */
                                   |* number of sectors per track (if known) *|
        u long
                 dk nsector;
                 dk name[DK MAXNAME]; |* name of device (if known) (e.g. "hd40r") *|
        char
};
#define DKIOCGPART
                          IOR(d, 2, struct dkpart) | * get partition information * |
The minidisk(8R) utility of sautil(8R) may be used to create and modify the minidisk directory
and 4.3/RT partition table. On the IBM 6152 Academic System, the fdisk(8R) utility of
sautil(8R) may be used to create and modify the master fixed disk boot record partition table.
```

SEE ALSO

fd(4), hd(4), sc(4), fdisk(8R), format(8R), minidisk(8R), sautil(8R)

"Chapter 12. IBM Predefined Device Driver--Reserved Cylinders on the Fixed Disk" in IBM RT PC Virtual Resource Manager Technical Reference, SV21-8013

"Chapter 3. Preparing Your Fixed Disk" in *Disk Operating System Version 3.30*, 80X0667, First Edition (April 1987).

This page intentionally left blank.

NAME

fd - diskette interface

SYNOPSIS

For the IBM RT PC:

controller fdc0 at iocc0 csr 0xf00003f2 priority 6 device fd0 at fdc0 drive 0

For the IBM 6152 Academic System:

controller fdc0 at iocc0 csr 0x000003f2 priority 6

device fd0 at fdc0 drive 0

device fd1 at fdc0 drive 1

DESCRIPTION

On the IBM RT PC, the fd device provides access to an IBM-PC/AT 1.2M diskette drive or an IBM-PC/AT 360k diskette drive through the diskette function of the IBM Personal Computer AT Fixed Disk and Diskette Adapter, the ESDI adapter, or the E-ESDI adapter. The disk drive uses 5.25-inch, double-sided, soft-sectored, and either double-density or high-capacity diskettes.

On the IBM 6152 Academic System, the fd device provides access to an System/2 1.4m diskette drive. The disk drive uses 3.5 inch, low or high density diskettes.

System/2 diskettes contain 80 cylinders with 2 tracks (heads) per cylinder. High density diskettes contain 18 sectors per track (2880 total sectors) and low density diskettes contain 9 sectors per track (1440 total sectors). The sector size is 512 bytes for both diskette types.

Standard PC-DOS diskettes (360k) contain 40x2 tracks, each with 9 sectors (for a total of 720 sectors). PC/AT high-capacity diskettes (1.2M) contain 80x2 tracks, each with 15 sectors (for a total of 2400 sectors). The sector size is 512 bytes for both diskette types.

When the device is opened, the density of the diskette currently in the drive is automatically determined. If there is no diskette in the drive, open will fail.

Raw I/O requests must start on a sector boundary, involve an integral number of complete sectors, and not go off the end of the diskette.

NOTES

Even though the storage capacity of diskettes is small, it is possible to make filesystems on them. For example, the command

% newfs /dev/fd0

makes a filesystem on fd0. (Using tar(1) instead of newfs(8) gives a more efficient utilization of the available space for file storage.)

A few ioctl(2) calls apply to the fd devices, and have the form

#include < machineio/fdio.h >
ioctl(fildes, code, arg)

int *arg;

The applicable codes are:

FDIOC_FORMAT

Format the diskette at the density specified by the *arg* argument (0 = 360k or 1 = 1.2M on the IBM RT PC, 0 = 720K or 1 = 1.4M on the IBM 6152 Academic System).

FDIOC GETDENS

Return the density of the diskette (0 = 360k or 1 = 1.2M on the IBM RT PC, 0 = 720K or 1 = 1.4M on the IBM 6152 Academic System).

FDIOC_RESET Reset the adapter.

You can open a drive for formatting using the flag FDO_FORMAT. If this flag is set, and the diskette in the drive has not been formatted, the open will successfully complete, but no other operations (except close and ioctl) may be performed.

There are older PC-DOS diskette formats that are not supported by fd(4).

ERRORS

The following errors may be returned by the driver:

[ENXIO] Nonexistent drive (on open); offset is too large or not on a sector boundary or byte count is not a multiple of the sector size (on read or write); bad (undefined) ioctl code. Drive not ready; usually because no diskette is in the drive or the drive door is open.

[EIO] A physical error other than "not ready", probably bad media or unknown format.

[EBUSY] Drive is being used to format a diskette (on open); drive is in use by someone else (on format ioctl).

[EBADF] No write access (on format), or wrong density; the latter can happen only if the diskette is changed without closing the device (i.e. calling *close*(2)).

FILES

/dev/fd[01] /dev/rfd[01]

SEE ALSO

tar(1), fdformat(8R), flcopy(8R), mkfs(8), newfs(8)

DIAGNOSTICS

For the IBM RT PC:

fd%d: hard error, trk = %d, sec = %d, Sr0 = 0x%b, Sr1 = 0x%b, Sr2 = 0x%b, Sr3 = 0x%b, Cyl = %d, Sec = %d, Head = %d, St = %d, state = %s. An unrecoverable error was encountered. The track and physical sector numbers, the device registers and the extended error status are displayed.

fd%d: state %d (reset). The driver entered an invalid state.

fd%d: timeout. Lost interrupt.

fd%d: write protected. The driver detected a drive containing a write-protected diskette. (On open only.)

fd%d: door open or hardware fault. Autodensity failed because the drive failed to interrupt. (On open only.)

fd%d: bad or unformatted diskette. Autodensity could not read the diskette. (On open only.)

For the IBM 6152 Academic System:

fd%d: hard error, BIOS error = 0x%b, trk = %d, Sec = %d, Head = %d, state = %s. An unrecoverable error was encountered. The Sytem/2 BIOS error code, track, sector, head and state are displayed.

fd%d: state %d (reset). The driver entered an invalid state.

fd%d: timeout. Lost interrupt.

fd%d: write protected. The driver detected a drive containing a write-protected diskette. (On open only.)

fd%d: door open or hardware fault. Autodensity failed due to an unrecoverable error. (On open only.)

fd%d: pc timeout. The device driver timed out waiting for the PC to accept a command.

BUGS

Diskettes written at 360k on a 1.2M drive are not guaranteed readable by a 360k drive on an IBM PC, PC/XT or PC/AT; they are readable by the 1.2M drive on a PC/AT.

This page intentionally left blank.

NAME

hd - hard disk interface

SYNOPSIS

For the IBM RT PC:
controller hdc0 at iocc0 csr 0xf00001f0 priority?
controller hdc1 at iocc0 csr 0xf0000170 priority?
disk hd0 at hdc0 drive 0
disk hd1 at hdc0 drive 1
disk hd2 at hdc? drive?
For the IBM 6152 Academic System:
controller hdc0 at iocc0 csr 0xffffffff priority 14
disk hd0 at hdc0 drive 0
disk hd1 at hdc0 drive 1

DESCRIPTION

On the IBM RT PC, this driver supports the fixed-disk function of the IBM Personal Computer AT Fixed Disk and Diskette Adapter, the Enhanced Small Device Interface Magnetic Media Adapter, and the Extended ESDI Magnetic Media Adapter. The diskette function of the adapter is supported by the fd(4) driver. The driver supports, at most, three hard disks on each of two adapters (for a total of six hard disks). Physical considerations limit the actual number of disks in a system to one or three, depending upon model. An Extended ESDI adapter will support up to three hard disks; the others are limited to at most two disks.

On the IBM 6152 Academic System, this driver supports System/2 fixed-disk adapters. The driver supports, at most, two hard disks on one adapter.

The "a," "b" and "g" partitions are by convention used for root, swap and user areas respectively. The "c" partition maps the physical area defined as the IBM/4.3 partition which includes disk space allocated for bootstraps and the replacement pool for bad blocks. On the IBM 6152 Academic System, the "h" partition is used to map the entire physical disk. This includes the first sector of the disk and may include areas used by other operating systems. The "h" partition is used by fdisk(8R) to access the master boot record partition table.

Files with minor device numbers 0 through 7 refer to partitions of drive 0; minor devices 8 through 15 refer to drive 1, and so on. The standard device names begin with "hd", followed by the drive number and then a letter a-h for partitions 0-7, respectively.

The block files access the disk via the system's normal buffering mechanism and may be read and written without regard to physical disk records. There is also a "raw" interface that provides for direct transmission between the disk and the user's read or write buffer. The names of the raw files conventionally begin with an extra r. A single read or write call results in exactly one I/O operation; therefore, raw I/O is more efficient when many words are transmitted. This is especially true for the System/2, as each I/O operation results in exactly one interrupt and no PIO transfer of the sector buffer is required. On the IBM RT PC, when using an Extended ESDI adapter data is transferred via DMA, even when using raw I/O. For raw I/O with the older IBM RT PC adapters, each 512-byte block still results in an interrupt and the PIO transfer of the sector buffer in or out of main memory.

In raw I/O, counts should be multiples of 512 bytes (a disk sector). Likewise, seek(2) calls should specify multiples of 512 bytes. For performance reasons, the user's buffer should be aligned on a fullword boundary when the raw device is used. Ensuring that the buffer does not cross a page boundary further improves performance.

DISK SUPPORT ON THE IBM RT PC

The driver reads the configuration record (cylinder 0, track 0, sector 2 using the convention of disk(4) to determine the geometry of the disk; this record is originally written by the format(8R) program and is further described in disk(4).

HD(4)

The origin and size (in sectors) of the default partitions on each drive are as follows. The character "?" stands for a drive number in the range 0-2.

hd40m 40 Mbyte hard disk partitions			
disk	start	length	capacity cylinders
hd?a	85	15884	7.8 MB 1 - 187
hd?b	15980	10032	4.9 MB 188 - 306
hd?c	0	87040	42.5 MB 0 - 1023
hd?d	26095	15884	7.8 MB 307 - 493
hd?f	41990	43945	21.5 MB 494 - 1010
hd?g	26095	59840	29.2 MB 307 - 1010
hd40r 40 Mbyte hard disk partitions			
disk	start	length	capacity cylinders
hd?a	119	15884	7.8 MB 1 - 134
hd?b	16065	10032	4.9 MB 135 - 219
hd?c	0	87227	42.6 MB 0 - 732
hd?d	26180	15884	7.8 MB 220 - 353
hd?f	42126	43911	21.4 MB 354 - 722
hd?g	26180	59857	29.2 MB 220 - 722
hd70m 70 MByte hard disk partitions			
disk	start	length	capacity cylinders
hd?a	136	15884	7.8 MB 1 - 117
hd?b	16048	33440	16.3 MB 118 - 363
hd?c	0	139264	68.0 MB 0 - 1023
hd?d	49504	15884	7.8 MB 364 - 480
hd?e	65416	55936	27.3 MB 481 - 892
hd?f	121448	16592	8.1 MB 893 - 1014
hd?g	49504	88536	43.2 MB 364 - 1014
hd70r 70 MByte hard disk partitions			
disk	start	length	capacity cylinders
hd?a	252	15884	7.8 MB 1 - 64
hd?b	16380	33440	16.3 MB 65 - 197
hd?c	0	142632	69.6 MB 0 - 565
hd?d	49896	15884	7.8 MB 198 - 261
hd?e	66024	55936	27.3 MB 262 - 483
hd?f	121968	19404	9.5 MB 484 - 560
hd?g	49896	91476	44.7 MB 198 - 560
hd70e 70 MByte hard disk partitions			
disk	start	length	capacity cylinders
hd?a	245	15884	7.8 MB 1 - 65
hd?b	16170	33440	16.3 MB 66 - 202
hd?c	0	142835	69.7 MB 0 - 582
hd?d	49735	15884	7.8 MB 203 - 267
hd?e	64190	55936	27.3 MB 268 - 496
hd?f	118580	19600	9.6 MB 497 - 576
hd?g	49735	91630	44.7 MB 203 - 576

It is unwise for all these special files to be present in one installation, because addresses overlap and protection becomes a sticky matter. The hd?a partition is normally used for the root file system, the hd?b partition as a paging area, and the hd?c partition for access to the entire disk, including boot and configuration information at the start of the disk and bad-block and diagnostic regions at the end of the disk. The safest way of using an entire disk as a single partition is NOT use of the hd?c partition. Instead, the *minidisk*(8R) standalone utility should be used to define a

partition that starts where the hd?a partition from the appropriate table above would (in other words, with the second cylinder) and ends where the hd?g would. If the hd?c partition is (unwisely) used for a filesystem, the following should be considered: The hd?c partitions cannot be used as filesystems with an hd70c disk due to the location of the hidden defect table. Other hd1c and hd2c partitions may be used as a filesystem provided that the bad-block forwarding table is not too large and the bad-block and diagnostic regions are not used. The newfs(8) program enforces this restriction. The hd0c partition cannot be used as a filesystem, because the root is normally on hd0a, swap is on hd0b, and /usr is on hd0g. In addition, the diagnostics write on hd0 following the bad-block table.

Standard partition tables are calculated using the diskpart(8) program. Non-standard disk partitions may be created using the minidisk(8R) menu selection of the sautil(8R) standalone utilities.

DISK SUPPORT ON THE IBM 6152 ACADEMIC SYSTEM

Disk geometry is provided by PS/2 initialization code at boot time. The number of cylinders per drive, tracks per cylinder (heads) and number of sectors per track is provided as well as the starting cylinder and size of the IBM/4.3 partition.

As the size of any IBM/4.3 physical partition is determined solely by the user at system configuration time, there can be no default partitionings. The partition information must be recorded on the disk itself. For reasons of code compatibility and support of existing disk formats, the VRM minidisk table is used for this purpose. Therefore, any disk with a IBM/4.3 physical partition must have a valid VRM minidisk table defining the existence of partitions.

These disk partitions are created using the minidisk(8R) menu selection of the sautil(8R) standalone utilities.

FILES

/dev/hd[0-2][a-h] block files raw files

SEE ALSO

fd(4), disk(4), diskpart(8), format(8R), minidisk(8R), newfs(8), sautil(8R)

"Chapter 3. Preparing Your Fixed Disk" in *Disk Operating System Version 3.30*, 80X0667, First Edition (April 1987).

DIAGNOSTICS

HD: adapter @ 0x%x TIMED OUT (0x%x & 0x%x != 0x%x). The adapter did not complete an operation in a reasonable length of time. Probable cause: defective adapter.

HD: Hard Disk I/O Error. An unrevoverable error occurred. If appropriate, numerous retry attempts have been made but all failed.

HD: Write Fault. Probable cause: bad drive.

HDINIT: Disk I/O Error CMD = 0x%x. The adapter reported an error while it was being initialized.

hd%d: %d blocks forwarded to zero - reformat required. Run the format(8R) utility.

hdc%d: diagnose = 0x%x ERROR. Probable cause: defective adapter.

hdc%d: lost interrupt(s). A timer watching the adapter detected no interrupt for an extended period of time while operation(s) were outstanding. Either the hd device driver or the adapter is failing. Recovery is attempted and the operation(s) are retried, which may cause the message to repeat.

hd: hd%d%c bn = %d Status = 0x%b,0x%b ErrA = 0x%x ErrB = 0x%x (IBM RT PC only) hd: hd%d%c bn = %d Err = 0x%b Status = 0x%b Cyl = %d Hd = %d Sect = %d SCT = %d Generic status messages which follow many of the more specific error messages. The second form appears only when the error occurred on an non-DMA operation, including all IBM 6152 Academic System operations. The first form is for DMA operations. If known, the specific drive, partition, and block number associated with the error are printed. The block number is relative to the start of the partition. The contents of various adapter registers are displayed.

hd%d: %s; interleave factor is %d to 1 (IBM RT PC)

hd%d: %s type (%d cyl, %d tracks, %d sectors); interleave factor is %d to 1 (IBM RT PC)

hd%d: %d mb allocated for IBM/4.3 (IBM 6152 Academic System)

hd%d: %d cylinders %d tracks %d sectors. (IBM 6152 Academic System)

Informational message displayed during autoconfiguration. If the drive type shown is "unknown" then the configuration record may be damaged.

For the IBM RT PC only:

HD: BAD INTERRUPT! ADAPTER NOT ACTIVE!

HD: BAD INTERRUPT! NO SUCH DRIVE!

HD: BAD INTERRUPT! DRIVE/QUEUE NOT ACTIVE!

HD: BAD INTERRUPT! INVALID STATUS!

HD: BAD INTERRUPT? UNEXPECTED STATUS!

These messages occur only when operating the EESDI adapter in DMA mode. Probable cause: adapter failure (or hd device driver bug).

HD: DMA CHANNEL ABUSE!!. All the hard disk DMA channels are in "exclusive" use so the hd driver was unable to get a DMA channel.

HD: HDXBREL: HELP! xbp = 0x%x NOT AN XBUF!

HD: HDDBUFDEO: bp 0x%x not on dp 0x%x queue!

HD: HDDDEVDEQ: dp 0x%x not on ic 0x%x queue!

These messages "can't happen." Probable cause: bug in hd device driver.

HD: HDDUNGO: dma free map FAILED!!. Probable cause: bug in dma device driver.

HD: OUT OF XBUFS! NHDXBUF = %d. The supply of buffer extensions has been exhausted. NHDXBUF is the number of buffer extensions configured into your kernel.

HDINT: DISK BUSY!? - INT. SWALLOWED (choke)

HDINT: NO DRQ (0x%x, 0x%x) - INT. TREATED AS WRITE REQUEST COMPLETE!!

HDINT: NO I/O IN PROGRESS!!

HDINT: NO STATUS - INT. IGNORED

HDINT: NO WRITE IN PROGRESS - INT. SWALLOWED (choke)

These messages occur only when DMA is not being used. Probable cause: adapter failure (or hd device driver bug).

hd%d: bad/missing configuration record. No valid configuration record was found at sector 1 during autoconfiguration. This probably means that the start of the disk has been overwritten. The disk will be unusable until the configuration record has been restored from a backup copy, or rebuilt with format(8R).

hd%d: no bad-block table

Indicates that no bad-block table was found at sector 8 during autoconfiguration. This probably means that the start of the disk had been overwritten and that the disk will not be usable until the bad-block information is restored from a backup image of the disk, or rebuilt by reformatting the entire disk.

hdc%d: ec level 0x%x microcode level 0x%x drive bits 0x%x. Informational message during autoconfiguration. This shows the adapter engineering change level, microcode level, and what drives it thinks are attached to it.

hdc%d: unknown adapter type 0x%x. During autoconfiguration it was noticed that the ROS was unable to identify the type of the specified disk adapter. This may later result in strange behavior.

hdswitch: nothing to start??. Probable cause: bug in hd device driver.

panic: hddbadblk. Probable cause: EESDI adapter failure (or hd device driver bug).

panic: hddgo bp
panic: hddgo channel
panic: hddgo dp- > b_actf
panic: hddgo ioaddr
panic: hddgo len
panic: hddgo len0

panic: hddgo pointers. These panics indicate bugs in the hd device driver.

panic: hddstart tews. Probable cause: bug in dma device driver.

panic: hdstrategy b bcount. Probable cause: bug in whatever called the hd device driver.

For the IBM 6152 Academic System

hd%d%c: soft ecc sn%d. A recoverable ECC error occurred on the specified sector of the specified disk partition. If it happens frequently, the pattern formed by the sectors where the errors occur should be checked to see if they suggest either a bad head or a bad spot on the disk.

BUGS

In raw I/O read(2) and write(2) truncate file offsets to 512-byte block boundaries, and write scribbles zeroes on the tail of incomplete blocks. Thus, in programs likely to access raw devices, read, write, and lseek(2) should always deal in 512-byte multiples.

In raw I/O, the buffer must be aligned on a fullword boundary.

DMA and overlapped seeks (and transfers) are done only with the Extended ESDI adapter.

This page intentionally left blank.

IBM5081(4)

NAME

ibm5081, mpel - IBM 5081 Mega Pel Display interface

SYNOPSIS

pseudo-device apasixteen

DESCRIPTION

The IBM 5081 Mega Pel Display interface is a 16- or 19-inch color CRT. It provides a 256 color out of a 4096 color pallette, all-points-addressable, bit-mapped display with 1,048,576 points on the screen (1024 pixels on each of 1024 displayable lines). A fast raster-operation processor is provided with built-in capability for a wide range of functions, including bit-block transfer, line draw, image copy/merge and image rotate. A graphics queue mechanism with synchronization and controlled branching allows pre-programmed graphics subroutines.

The display adapter is a dual PC/AT card installed in the I/O bus as a sixteen-bit device. The bit-map is not directly addressable. There are 2-641c shared banks at f4c00000 and f4c10000 used to share fonts and communicate with the microcode.

The display operates as /dev/ttympel in glass tty (the default) mode and /dev/mpel in window-manager mode:

- Glass tty mode initialization consists of the downloading of a character font and the microcode must already be loaded (boot does this). In this mode, the display driver emulates a smart terminal, similar to an IBM 3101, and can be /dev/console.
- In window-manager mode, a user-level process, such as a window manager, can directly control the display device hardware, loading control programs, issuing graphics orders, etc. When a process opens /dev/mpel, output intended for /dev/ttympel is buffered by the kernel until later (see bufemul(4)). The process must then open /dev/bus to access the display control memory areas for manipulation by the user program if bit 0x8 is not set in the minor device number. Glass tty mode is reentered when /dev/mpel is closed. NOTE: The kernel cannot reload the microcode.

NOTES

This adapter can also drive an IBM 5082 projection display.

ERRORS

The following errors can be returned by the interface:

[ENODEV] Nonexistent display (on open, close, read, write, or ioctl);

Unavailable display (on open): user processes are denied access to this display (see consoles(5), setscreen(8)).

[EIO] Made an attempt to close a display device that was not open.

NOTES

Ibm5081 is not supported on the IBM 6152 Academic System.

FILES

/dev/mpel /dev/ttympel /dev/console /usr/lrb/mpel microcode files

SEE ALSO

bufemul(4), bus(4), cons(4), ibm5151(4), ibm5154(4), ibm6153(4), ibm6154(4), i

"IBM/4.3 Console Emulators" in Volume II, Supplementary Documents

IBM5081(4)

BUGS

Access to the PC/AT I/O and memory busses through /dev/apa16, when bit 0x8 is set in the minor device number, is not limited to the apa16 addresses.

The microcode is over 100K and cannot be loaded by the kernel.

IBM5151(4)

NAME

ibm5151, mono - IBM 5151 Monochrome Display interface

SYNOPSIS

pseudo-device mono

DESCRIPTION

The monochrome display is a standard IBM Personal Computer AT Monochrome Display and display adapter. The display adapter is accessed through a memory-mapped I/O address space. The display buffer is at address 0xf40b0000. Opening |dev|mono grants direct access to the display buffer and the monochrome registers in the I/O map if bit 0x8 is set in the minor device number. The registers are at the addresses given in the IBM PC/AT Technical Reference Manual, offset by 0xf0000000.

Each character on the display has two bytes in the buffer; one contains the character, the other the attributes. The attributes available are reverse video, intense mode, blink mode and underline mode. These are available through escape sequences sent to the display (see *ibmemul(4)*). All other functions are provided through software terminal emulation, including either 24-line (plus status) mode or 25-line mode, forward and reverse scrolling and cursor positioning.

The display is normally controlled by *ibmemul*, and accessed as /dev/ttymono. (See cons(4)). It may also be opened and manipulated as /dev/mono and /dev/bus for nonstandard use.

DIAGNOSTICS

None.

FILES

/dev/console /dev/ttymono /dev/mono

SEE ALSO

bus(4), cons(4), ibm6153(4), ibm6154(4), ibm6155(4), ibmaed(4), ibmemul(4), kbdemul(4), tty(4) "IBM/4.3 Console Emulators" in Volume II, Supplementary Documents

BUGS

Access to the PC/AT I/O and memory busses through /dev/mono, when bit 0x8 is set in the minor device number, is not limited to the mono addresses.

This page intentionally left blank.

IBM5154(4)

NAME

ibm5154, ega - IBM 5154 Enhanced Graphics Display interface

SYNOPSIS

pseudo-device ega

DESCRIPTION

The ega is a standard IBM Personal Computer AT Enhanced Graphics Display and display adapter. The display adapter is accessed through a memory-mapped I/O address space. The display buffer is at address 0xf40b8000. Opening |dev|ega grants direct access to the display buffer and the monochrome registers in the I/O map if bit 0x8 is set in the minor device number. The registers are at the addresses given in Volume II of the IBM RT PC Technical Reference Manual, offset by 0xf0000000.

Each character on the display has two bytes in the buffer; one contains the character, the other the attributes. The attributes available are eight independently selectable colors for foreground and background, blink mode, and intense mode. There is no underline mode for the ega. These are available through escape sequences sent to the display (see *ibmemul(4)*). All other functions are provided through software terminal emulation, including either 24-line (plus status) mode or 25-line mode, forward and reverse scrolling and cursor positioning.

The display is normally controlled by *ibmemul*, and accessed as /dev/ttyega. (See cons(4)). It may also be opened and manipulated as /dev/ega and /dev/bus for nonstandard use. The default color table for the ega color table is:

0 black
1 blue
2 green
3 cyan
4 red
5 violet
6 yellow

white

The ega has eight color table entries which can select from a palette of eight colors.

NOTES

Ibm5154 is not supported on the IBM 6152 Academic System.

DIAGNOSTICS

None.

FILES

/dev/console /dev/ttymono /dev/mono

SEE ALSO

bus(4), cons(4), ibm5151(4), ibm6153(4), ibm6154(4), ibm6155(4), ibmaed(4), ibmemul(4), kbdemul(4), tty(4)

"IBM/4.3 Console Emulators" in Volume II, Supplementary Documents

BUGS

Access to the PC/AT I/O and memory busses through |dev|ega, when bit 0x8 is set in the minor device number, is not limited to the mono addresses.

IBM5154(4) IBM5154(4)

THE FOLLOWING APPLIES ONLY TO THE IBM 6152 ACADEMIC SYSTEM: SYNOPSIS

pseudo-device ega

DESCRIPTION

The IBM Enhanced Graphics Adapter (EGA) is a graphics controller that supports both color and monochrome direct drive displays in a variety of modes. Support for the IBM PC/AT includes only 80 x 25 alphanumeric modes for the Monochrome Display, the Color Graphics Display and the Enhanced Color Display. The display adapter is accessed through a memory-mapped I/O address space. The display buffer is at address 0x000b0000 for the mono display and at address 0x000b8000 for the color and enhanced color displays. Opening |dev|ega grants direct access to the display buffer and the registers in the I/O map. The registers are at the addresses given in the IBM Technical Reference Options and Adapters IBM Enhanced Graphics Adapter section, offset by a run-time variable pcif_io_b. (See George's stuff.)

Each character on the display has two bytes in the buffer; one contains the character, the other the attribute. In mono mode, the attributes available are reverse video, intense mode, blink mode and underline mode. In color or enhanced color mode, the foreground and background colors can be changed. These are available through escape sequences sent to the display (see below). All other functions are provided through software terminal emulation, including foreward and reverse scrolling and cursor positioning.

In mono mode the escape sequences recognized by the display software are from the same termcap entry (in /etc/termcap) as for the ibm5151 (mono) "ibmmono". In color mode the escape sequences are from the termcap entry "ibmega" and are as follows:

Cursor up	
Non-desructive space	
Home cursor	
Clear to end of line	
Clear to end of screen	
Clear screen	
Insert line	
Delete line	
Change foreground color	
Change background color	
X Y position cursor	
Save cursor position	
Restore cursor position	
Status line on/off	

The color support for the Enhanced and Color displays is available from the PC/AT keyboard with two function keys: F3 - change foreground or F4 - change background. The stty option of -ctlecho should be set before executing either of these functions.

FILES

/dev/console /dev/ega

SEE ALSO

stty(1), cons(4), ibm5151(4), kbdemul(4), tty(4)

Installation instructions for the "IBM Enhanced Graphics Adapter" in the IBM Technical Reference Options and Adapters

DIAGNOSTICS

If the switches on the adapter are not set for a supported configuration, the probe for the adapter during autoconf will fail and the screen attached to the adapter will be unavailable to the system.

IBM5154(4)

BUGS

Support should be fully implemented to use the full capabilities of the adapter.

IBM6153(4) IBM6153(4)

NAME

ibm6153, apa8 - IBM 6153 Advanced Monochrome Graphics Display interface

SYNOPSIS

pseudo-device apaeight

DESCRIPTION

The IBM 6153 Advanced Monochrome Graphics Display is a 12-inch CRT with gray-white phosphor, driven at 92 Hz interlaced. It provides a monochrome, all-points-addressable, bitmapped display with 393,216 points on the screen (720 displayable pixels on each of 512 displayable lines). All pixels are directly accessible by the CPU. The display adapter provides hardware assist features including a write mask to protect bit fields within a byte, a barrel shifter to rotate bits within a byte, and a logic unit to combine source bytes before they are written into the bit

The display adapter is a single PC/AT card installed in the I/O bus as a sixteen-bit device. The display appears to the system as two separate memory areas: a 128k block of system memory (beginning at 0xf4d00000), and 16 bytes of I/O space (addressed from 0x160 through 0x16f). The 128k block defines both the visible frame buffer and the hidden, off-screen memory area. For each of the 512 scan lines, the first 90 bytes (720 pixels) are visible; the last 38 bytes (304 pixels) are hidden. The 16 bytes of I/O space access the display adapter's control registers.

There are several different read and write modes plus several different masks and operations described in the IBM RT PC Hardware Technical Reference Volume II, Advanced Monochrome Graphics Display Adapter. Below is the interaction of these modes and user read/writes to display memory. Remember all read and writes are shorts (16 bit values).

```
Definitions:
*addr
        The "high" byte addressed by the user.
*addr+
        The "low" byte addressed by the user. Note this could be either left, right, up, or down
        from addr depending on the settings of the Increment/Decrement (Address Counter
        Mode) and X/Y (Address counter Stepping) in the Data Control Register.
        The "high" byte specified (or read) by the user.
sy 1
sy2
        The "low" byte specified (or read) by the user.
dm1
        data mask 1
dm2
        data mask 2
wm1
        write mask 1
wm2
        write mask 2
d1
        internal date register 1
d2
        internal date register 2
d3
        internal date register 3
```

x is rotated the amount specified by the rotate count in the Data Control Register.

IBM6153(4)

```
func(a,b)
       The output of the logical function unit operating on a and b. The actual function is
       specified by the Logical Unit Function Control bits in the Data Control Register.
mode
       The Memory Mode bits in the Data Control Register.
       8 bit and.
&
       8 bit or.
       8 bit ones compliment.
Modes:
System Read Operation (load half, mode = = any)
               d1 = *addr
               d2 = *addr +
               sy1 = *addr (only if mode = = 00)
               sy2 = *addr + (only if mode = = 00)
System Write Operation (store half, mode = = 00)
               *addr = (wm1 & *addr) | (~wm1 & func(sy1 & dm1,rot(d2) & dm2))
               *addr + = (wm2 & *addr +) (~wm2 & func(sy2 & dm1,rot(d1) & dm2))
Overlay Write Operation (store half, mode = 01)
               *addr = (wm1 \& *addr) | (~wm1 \& func(sy1 \& dm1,rot(d2) \& dm2))
               *addr + = (wm2 & *addr +) ( wm2 & func(sy2 & dm1,rot(d1) & dm2))
               wm1 = sy1
               wm2 = sy2
Adapter Write Operation (store half, mode = = 10)
               *addr = (wm1 \& *addr) | (~wm1 \& func(d3 \& dm1,rot(d2) \& dm2))
               *addr + = (wm2 & *addr + ) | (~wm2 & func(rot(d2)&dm1,rot(d1)&dm2))
               d3 = rot(d2)
               d2 = d1
Automatic Read/Write Operation (store half, mode = = 11)
               *addr = (wm1 \& *addr) | (``wm1 \& func(d3 \& dm1,rot(d2) \& dm2))
               *addr + = (wm2 & *addr +) | (~wm2 & func(rot(d2)&dm1,rot(d1)&dm2))
               d3 = d2
               d2 = d1
```

The display operates as /dev/ttyapa8 in glass tty (the default) mode and /dev/apa8 in window-manager mode:

- Glass tty mode initialization consists of the downloading of a character font into the adapter card, followed by a cursor home and screen clear. In this mode, the display driver emulates a smart terminal, similar to an IBM 3101, and can be /dev/console.
- In window-manager mode, a user-level process, such as a window manager, can directly control the display device hardware, loading picture data, accessing display buffers, etc. When a process opens /dev/apa8, output intended for /dev/ttyapa8 is buffered by the kernel until later (see bufemul(4)). The process must then open /dev/bus to access the display and control memory areas for manipulation if bit 0x8 is not set in the minor device number. Glass tty mode is reentered when /dev/apa8 is closed.

DIAGNOSTICS

None.

d1 = *addr

養

IBM6153(4)

ERRORS

The following errors can be returned by the interface:

[ENODEV] Nonexistent display (on open, close, read, write, or ioctl);

Unavailable display (on open): user processes are denied access to this display (see consoles(5), setscreen(8)).

[EIO]

Made an attempt to close a display device that was not open.

[EBUSY] The display has already been opened by a user process.

NOTES

Ibm6153 is not supported on the IBM 6152 Academic System.

FILES

/dev/apa8 /dev/ttyapa8 /dev/console

SEE ALSO

bufemul(4), bus(4), cons(4), ibm5081(4), ibm5151(4), ibm5154(4), ibm6154(4), ibm6155(4), ibm8514(4), ibmaed(4), ibmemul(4), kbdemul(4), vga(4)
"IBM/4.3 Console Emulators" in Volume II, Supplementary Documents

BUGS

Access to the PC/AT I/O and memory busses through |dev|apa8, when bit 0x8 is set in the minor device number, is not limited to the apa8 addresses.

IBM6154(4) IBM6154(4)

NAME

ibm6154, apa8c - IBM 6154 Advanced Color Graphics Display interface

SYNOPSIS

pseudo-device apaeightc

DESCRIPTION

The IBM 6154 Advanced Color Graphics Display is a 14-inch CRT with a 0.3mm dot-pitch shadow mask, driven at 92 Hz interlaced. It provides a color, all-points-addressable, bit-mapped display with 393,216 points on the screen. The display adapter provides a 256k bit-map translated to 720 displayable pixels on each of 512 displayable lines, 4 bits per pixel. The bit-map is organized as 4 planes (64k per plane). Each pixel can take one of 16 colors selected from a 64-color palette. All pixels are directly accessible by the CPU.

The display adapter provides hardware assist features including a write mask to protect bit fields within a byte, a barrel shifter to rotate bits within a byte, and a logic unit to combine source bytes before they are written into the bit map. In addition, a plane select register and foreground/background register are provided as a means to select individual planes or all planes for update with pre-programmed foreground/background color data. The color palette is provided with a 4-bit to 6-bit lookup memory. Each of the 16 entries contains two bits per primary color (red, red intense, green, green intense, blue, blue intense). The default values for the apa8c color table are:

0	black	8	dark gray
1	blue	9	dark blue
2	green	10	dark green
3	cyan	11	orange
4	red	12	dark red
5	violet	13	pink
6	yellow	14	yellow-orange
7	light gray	15	white

The display adapter is a single PC/AT card installed in the I/O bus as a sixteen-bit device. To be upward-compatible with the IBM 6153, the display appears to the system as two separate memory areas: a 128k block of system memory (beginning at 0xf4d20000), and 16 bytes of I/O space (addressed from 0x150 through 0x15f). The 128k block defines both the visible frame buffer and the hidden, off-screen memory area. For each of the 512 scan lines, the first 90 bytes (720 pixels) are visible; the last 38 bytes (304 pixels) are hidden. The 16 bytes of I/O space access the display adapter's control registers.

There are several different read and write modes plus several different masks and operations described in the IBM RT PC Hardware Technical Reference Volume II, Advanced Color Graphics Display Adapter. Below is the interaction of these modes and user read/writes to display memory. Remember all read and writes are shorts (16 bit values).

Definitions:

*addr The "high" byte addressed by the user.

*addr+

The "low" byte addressed by the user. Note this could be either left, right, up, or down from addr depending on the settings of the Increment/Decrement (Address Counter Mode) and X/Y (Address counter Stepping) in the Data Control Register.

syl The "high" byte specified (or read) by the user.

sy2

The "low" byte specified (or read) by the user.

IBM6154(4)

```
dm1
        data mask 1
dm2
       data mask 2
wm1
       write mask 1
wm2
        write mask 2
dl
       internal date register 1
d2
       internal date register 2
d3
       internal date register 3
       x is rotated the amount specified by the rotate count in the Data Control Register.
rot(x)
func(a,b)
       The output of the logical function unit operating on a and b. The actual function is
       specified by the Logical Unit Function Control bits in the Data Control Register.
       The Memory Mode bits in the Data Control Register.
&
       8 bit and.
       8 bit or.
       8 bit ones compliment.
Modes:
System Read Operation (load half, mode = = any)
               d1 = *addr
               d2 = *addr +
               syl = *addr (only if mode = = 00)
               sy2 = *addr + (only if mode = = 00)
System Write Operation (store half, mode = = 00)
               *addr = (wm1 & *addr) | (~wm1 & func(sy1 & dm1,rot(d2) & dm2))
               *addr + = (wm2 & *addr +) | (~wm2 & func(sy2 & dm1,rot(d1) & dm2))
Overlay Write Operation (store half, mode = 01)
               *addr = (wm1 \& *addr) | (~wm1 \& func(sy1 \& dm1,rot(d2) \& dm2))
               *addr + = (wm2 & *addr +) ( wm2 & func(sy2 & dm1,rot(d1) & dm2) 
               wm1 = sy1
               wm2 = sy2
Adapter Write Operation (store half, mode = = 10)
               *addr = (wm1 \& *addr) | (~wm1 \& func(d3 \& dm1,rot(d2) \& dm2))
               *addr + = (wm2 & *addr +) ( wm2 & func(rot(d2) & dm1, rot(d1) & dm2) )
               d3 = rot(d2)
               d2 = d1
Automatic Read/Write Operation (store half, mode = 11)
               *addr = (wm1 & *addr) | (~wm1 & func(d3 & dm1,rot(d2) & dm2))
               *addr + = (wm2 & *addr +) | (~wm2 & func(rot(d2)&dm1,rot(d1)&dm2))
               d3 = d2
```

IBM6154(4)

```
d2 = d1
d1 = *addr
```

The display operates as /dev/ttyap8c in glass tty (the default) mode and /dev/apa8c in window-manager mode:

- Glass tty mode initialization consists of the downloading of a character font into the adapter card, followed by a cursor home and screen clear. In this mode, the display driver emulates a smart terminal, similar to an IBM 3101, and can be /dev/console.
- In window-manager mode, a user-level process, such as a window manager, can directly control the display device hardware, loading picture data, accessing display buffers, etc. When a process opens /dev/apa8c, output intended for /dev/ttyap8c is buffered by the kernel until later (see bufemul(4)). The process must then open /dev/bus to access the display and control memory areas for manipulation if bit 0x8 is not set in the minor device number. Glass tty mode is reentered when /dev/apa8c is closed.

DIAGNOSTICS

None.

ERRORS

The following errors can be returned by the interface:

[ENODEV] Nonexistent display (on open, close, read, write, or ioctl);

Unavailable display (on open): user processes are denied access to this display (see consoles(5), setscreen(8)).

[EIO] Made an attempt to close a display device that was not open.

[EBUSY] The display has already been opened by a user process.

NOTES

Ibm6154 is not supported on the IBM 6152 Academic System.

FILES

/dev/apa8c /dev/ttyap8c /dev/console

SEE ALSO

bufemul(4), bus(4), cons(4), ibm5081(4), ibm5151(4), ibm5154(4), ibm6153(4), ibm6155(4), ibm8514(4), ibmaed(4), ibmemul(4), kbdemul(4), tty(4), vga(4)
"IBM/4.3 Console Emulators" in Volume II, Supplementary Documents

BUGS

Access to the PC/AT I/O and memory busses through /dev/apa8c, when bit 0x8 is set in the minor device number, is not limited to the apa8c addresses.

IBM6155(4) IBM6155(4)

NAME

ibm6155, apa16 - IBM 6155 Extended Monochrome Graphics Display interface

SYNOPSIS

pseudo-device apasixteen

DESCRIPTION

The IBM 6155 Extended Monochrome Graphics Display is a 15-inch CRT with gray-white phosphor, driven at 60 Hz non-interlaced. It provides a monochrome, all-points-addressable, bit-mapped display with 786,432 points on the screen (1024 pixels on each of 768 displayable lines). All pixels are directly accessible by the workstation's CPU. A fast raster-operation processor is provided with built-in capability for bit-block transfer, line draw, image copy/merge and image rotate. A graphics queue mechanism with synchronization and controlled branching allows preprogrammed graphics subroutines. A 48x64-bit hardware cursor provides instant cursor operations without disruption of display data.

The display adapter is a single PC/AT card installed in the I/O bus as a sixteen-bit device. The display appears to the system as two separate memory areas: a 128k block of system memory, and 20 bytes of I/O space (addressed from 0xd10 through 0xd2f, and 0x6f3). The 128k block defines both the visible frame buffer (addressed from 0xf4d80000 through 0xf4d97ffe) and the hidden, off-screen memory area (addressed from 0xf4d98000 through 0xf4d9fffe). The 20 bytes of I/O space access the display adapter's control registers.

The display operates as |dev|ttyap16 in glass tty (the default) mode and |dev|apa16 in window-manager mode:

- Glass tty mode initialization consists of the downloading of a character font and certain graphics subroutines into the adapter card, followed by a cursor home and screen clear. In this mode, the display driver emulates a smart terminal, similar to an IBM 3101, and can be /dev/console.
- In window-manager mode, a user-level process, such as a window manager, can directly control the display device hardware, loading control programs, accessing display buffers, etc. When a process opens /dev/apa16, output intended for /dev/ttyap16 is buffered by the kernel until later (see bufemul(4)). The process must then open /dev/bus to access the display control memory areas for manipulation by the user program if bit 0x8 is not set in the minor device number. Glass tty mode is reentered when /dev/apa16 is closed.

DIAGNOSTICS

None.

ERRORS

The following errors can be returned by the interface:

[ENODEV] Nonexistent display (on open, close, read, write, or ioctl);

Unavailable display (on open): user processes are denied access to this display (see consoles(5), setscreen(8)).

[EIO] Made an attempt to close a display device that was not open.

[EBUSY] The display has already been opened by a user process.

NOTES

Ibm6155 is not supported on the IBM 6152 Academic System.

FILES

/dev/apa16 /dev/ttyap16 /dev/console IBM6155(4)

SEE ALSO

bufemul(4), bus(4), cons(4), ibm5081(4), ibm5151(4), ibm5154(4), ibm6153(4), ibm6154(4), ibm8514(4), ibmaed(4), ibmemul(4), kbdemul(4), tty(4), vga(4)
"IBM/4.3 Console Emulators" in Volume II, Supplementary Documents

BUGS

Access to the PC/AT I/O and memory busses through /dev/apa/6, when bit 0x8 is set in the minor device number, is not limited to the apa16 addresses.

IBM8514(4) IBM8514(4)

NAME

ibm8514 - IBM 8514/A Display adapter for the ibm8503, ibm8513, ibm8514, ibm8604

SYNOPSIS

pseudo-device ibmybarmmmdxiv

DESCRIPTION

The IBM 8514/A display adapter is an optional adapter card for the IBM PS/2. It is capable of driving all displays available for the VGA (see vga(4)).

In function mode, the IBM 8514 adapter with the IBM 8514 Display can display 1024 x 768, 256 colors out of a palette of 256,000. Smaller displays provide lower resolutions. In VGA emulation mode, the ibm8514 adapter mirrors what is displayed on the VGA screen.

In window manager mode, bufemul(4) supplies an ioctl that initializes the 8514 to advanced function mode. Closing the 8514 reenables VGA emulation mode. There is no documented interface to the 8514 advanced function mode.

There is no glass tty support. Data written to /dev/tty85/4 is lost.

DIAGNOSTICS

None.

FILES

```
/dev/console
/dev/ttyvga
/dev/vga
/dev/tty8514
/dev/ibm8514
```

SEE ALSO

```
bufemul(4), cons(4), ibmemul(4), kbdemul(4), tty(4) "IBM/4.3 Console Emulators" in Volume II, Supplementary Documents
```

BUGS

There is no documented advanced function interface. There is no glass tty support. (Ibm8514 uses VGA emulation mode.)

IBMAED(4)

NAME

ibmaed, aed - IBM Academic Information Systems experimental display interface

SYNOPSIS

pseudo-device aed

DESCRIPTION

The IBM Academic Information Systems experimental display is a monochrome, all-points-addressable, bit-mapped display with 819,200 points on the screen. It is driven by an adapter with an on-board processor. The processor executes a microprogram loaded by the host into the display's control store. The microprogram controls the interface between host and display. The host communicates with the microprogram through an area of read/write shared RAM. This memory starts at address 0xf40a4000 and is organized as 4k of 8-bit bytes.

The display operates as |dev|ttyaed in glass tty mode (the default) and |dev|aed in window-manager mode:

- During glass tty mode initialization, the display driver loads the control store with a default microprogram which includes a built-in font. This is followed by a cursor home and screen clear. In glass tty mode, the display driver emulates a smart terminal, similar to an IBM 3101, and can be /dev/console.
- In window-manager mode, a user-level process, such as a window manager, can load its own microprogram into the display's control store, and read or write to the display's communications memory. When a process opens /dev/aed, output intended for /dev/ttyaed is buffered by the kernel until later (see bufemul(4)). The process must then open /dev/bus to access the control store and communications memory areas when bit 0x8 is not set in the minor device number. Glass tty mode is reentered when /dev/aed is closed.

In glass tty mode, the aed microcode will interpret certain escape sequences to control display behavior. The recognized escape sequences are listed below. < Esc > indicates the escape character 0x1b. Home refers to the cursor position in the upper left corner of the display. The notation (n excess 0x20) means character n will be used as a binary value less 0x20. For example, to encode a binary 4, add 0x20. The ASCII character 0x24 is '\$'.

- < Esc > 0 Begin/end stand-out mode. Characters are displayed in reversed color.
- < Esc > 1 Begin/end underline mode. Characters are displayed as underlined.
- < Esc > 2 Begin/end bold mode. Characters are displayed as bold (thicker than normal).
- <Esc > 3 Begin/end special mode. Enables direct access to the status line, and makes it the last line of the screen. While in special mode, a delete-to-end-of-line or delete-to-end-of-screen will replace deleted characters with blanks having the current attributes (underlined, reversed). Placing a character in the lower-right corner will cause a wrap to upper-left corner instead of a scrolling action. This mode provides specialized support to the display driver, and is not intended for normal use. Ending special mode restores normal operation.
- Status line processing. Beginning in column (n excess 0x20), write characters to the status line. Status line processing stops when the right margin is reached or when a null character (binary 0x00), < Esc > @, or carriage return is encountered.
- < Esc > 8n Retrieve the contents of display line (n excess 0x20). The character string is stored beginning at host memory location 0xf40a4100. Character attributes are stored at location 0xf40a4180.
- < Esc > 9ftd Copy lines from (f excess 0x20) through (t excess 0x20) to the destination area (d excess 0x20). Overlapping copy is not prevented. The order in which from and to

IBMAED(4) IBMAED(4)

Move the cursor up one line (up-arrow).

occur controls the direction of the copy. If from has a larger value than to, the destination line will be at the bottom of the copied image. This function is provided in support of the special mode (see above), but may be used independently.

· LSC · A	wove the cursor up one line (up-arrow).	
< Esc $>$ B	Move the cursor down one line (down-arrow):	
< Esc $>$ C	Move the cursor right one character (right-arrow).	
< Esc > D	Move the cursor left one character (left-arrow).	
< Esc $>$ E	Erase the status line.	
< Esc > H	Move the cursor to home.	
< Esc > I	Delete from current position to end of current line.	
< Esc > J	Delete from current position to end of display (bottom-right corner). The status line is left intact.	
< Esc > K	Move the cursor to home and erase the display.	
< Esc $>$ L	Move the cursor to home and erase the display.	
< Esc $>$ N	Insert a line below the current one. Scroll the remaining display lines down.	
< Esc > O	Delete the current line. Scroll the remaining display lines up.	
< Esc $>$ Pc	Insert character c at the current position. Shift the remaining characters on the current line to the right.	
< Esc $>$ S	Reverse the display color (background and characters).	

< Esc > Tc
Change the hardware tab setting to (c excess 0x30). Acceptable values are binary 0x1 through 0x9, represented by the characters '1' through '9'. When microcode is reloaded, the value is initialized to 8.

Delete the character at the current position. Shift the remaining characters on the

- < Esc > Q Delete the character at the current position. Shift the remaining characters on the current line to the left.
- < Esc > Yrc Position the cursor at row (r excess 0x20), column (c excess 0x20).
- < Esc > @ Terminate status line.

A few of these sequences are useful for entry from the keyboard, including begin/end standout mode, begin/end underline mode, begin/end bold mode, and reverse display color. For example, if the display is accidently left in standout mode, the user may correct it by entering echo "< Esc > 0".

The aed may also use *ibmemul*(4), in which case the above escape sequences would be replaced by the *ibmemul* escape codes.

A convenient way to reset all modes (especially useful if the display has become confused by trying to print a binary file) is to enter echo > /dev/aed.

As stated above, opening |dev|aed and |dev|bus will buffer glass tty output and gain access to the display hardware addresses. However, if the output emulator is changed to the aed emulator, support will be available for normal device system calls. See aedemul(4).

NOTES

Ibmaed is not supported on the IBM 6152 Academic System.

FILES

/dev/console /dev/aed /dev/ttyaed

< Esc > A

IBMAED(4)

SEE ALSO

aedemul(4), bus(4), cons(4), ibm5081(4), ibm5151(4), ibm5154(4), ibm6153(4), ibm6154(4), ibm6155(4), ibm8514(4), ibm_emul(4), kbdemul(4), stdemul(4), tty(4), vga(4), aedtest(8) "IBM/4.3 Console Emulators", in Volume II, Supplementary Documents

BUGS

Access to the PC/AT I/O and memory busses through /dev/aed, when bit 0x8 is set in the minor device number, is not limited to the aed addresses.

IBMEMUL(4)

NAME

ibmemul - IBM 3101 emulator

SYNOPSIS

pseudo-device apasixteen pseudo-device apaeightc pseudo-device apaeight pseudo-device ega pseudo-device mono

DESCRIPTION

Ibmemul is an output emulator which interprets a superset of escape sequences recognized by the IBM 3101 ASCII terminal. This emulator is the default emulator for the IBM 5151 Monochrome Display (mono), IBM 5154 Enhanced Graphics display (ega), IBM 6153 Advanced Monochrome Graphics Display (apa8), IBM 6154 Advanced Color Graphics Display (apa8c), and the IBM 6155 Extended Monochrome Graphics Display (apa16). If none of these devices is configured, ibmemul is not included.

Below is a list of the recognized escape sequences:

<Esc $>$ A	Cursor up
<Esc $>$ C	Non-destructive space
<Esc $>$ H	Home cursor
<Esc $>$ I	Clear to end of line
<Esc $>$ J	Clear to end of screen
<Esc $>$ K	Clear screen
<Esc $>$ L	Insert line
<Esc $>$ M	Delete line
<Esc $>$ W	Start underline
<Esc $>$ Y	X Y position cursor
<Esc $>$ Z	High intensity
<Esc $>$ j	Save cursor position
<Esc $>$ k	Restore cursor position
<Esc $>$ p	Reverse on
<Esc $>$ q	Reverse off
<Esc $>$ s	Status line on/off
<Esc $>$ w	Stop underline
<Esc $>$ z	Low intensity
<Esc $>$ V	Increment the color table value of the current background color
<Esc $>$ U	Increment the color table value of the current foreground color
<Esc $>$ v	Decrement the color table value of the current background color
<Esc $>$ u	Decrement the color table value of the current foreground color
<Esc $>$ R	Reverse the current foreground and background colors
<Esc $>$ G	Start blink
<Esc $>$ E	End blink
<Esc $>$	Increment display mode (VGA only)
<Esc $>$ fhex;	Select color table entry hex for the foreground
< Esc $>$ bhex;	Select color table entry hex for the background
<Esc $>$ F	Save the current foreground and background entries (Note: This
	does not save the color table values associated with these entries.)
<Esc $>$ B	Restore saved foreground and background entries.
< Esc > Tentry;red;gre	en;blue;
	C-4 Alice - 1 - 0 A - 1 - 1 0 A - 0 0 A - 0 A

Set the color table entry entry (specified in hex) to the color specified by red, green, and blue (also specified in hex)

IBMEMUL(4)

Oversize values for color table entries are ignored when specified in the <Esc> T command. Only the significant low order bytes are used for oversize values specified by the <Esc> f and <Esc> b commands. All values for red, green, and blue are normalized to the 32-bit hex value (i.e. 1 becomes 10000000, f becomes f0000000, 3f becomes 3f0000000, and 001 becomes 00100000). Only the high order bits are significant (the number of bits depends on the number of colors the display may use.)

Monochrome displays have two color table entries which may not be set to the same color. Thus $\langle Esc \rangle V$, $\langle Esc \rangle V$, and $\langle Esc \rangle V$ are to the same color table entry.

To return to *ibmemul* after switching to another emulator, or if the console device you opened does not use *ibmemul* as a default, do the following:

```
#include < machinecons/screen_conf.h >
...
int fd, output_emul;
char *console; /* set to some console device */
...
ifd = open (console, O_RDWR);
...
output_emul = E_IBMOUTPUT;
ioctl(fd,EOSETD,&output_emul);
```

NOTES

< Esc > S and < Esc > < change the screen size. They issue a SIGWINCII to indicate the new screen size.

FILES

/dev/console /dev/ttyap16 /dev/ttyap8c /dev/ttyapa8 /dev/ttyega /dev/ttymono

SEE ALSO

cons(4), ibm5151(4), ibm5154(4), ibm6153(4), ibm6154(4), ibm6155(4), ibmaed(4), kbdemul(4), stdemul(4)

"IBM/4.3 Console Emulators" in Volume II, Supplementary Documents

DIAGNOSTICS

None.

BUGS

There are not separate escape sequences to turn the status line on and off.

NAME

kbdemul - default keyboard emulator

DESCRIPTION

The keyboard adapter returns scan codes which the keyboard emulator translates into ASCII characters. This translation is done through table lookup; pf(1) allows users to change the translation tables so as to redefine the ASCII characters generated by particular keystrokes.

There are two sets of translation tables: a standard set not normally changed (see keyboard codes(5)), and a programmable set the user may freely change as desired (see pf).

In addition to generating ASCII characters, the keyboard emulator interface can invoke special functions (swapping of key definitions, etc.; see below). Most of these special functions are bound to the three keys labeled < Print-Screen/SysRq>, < Scroll-Lock>, and < Pause/Break>. The normal use of these keys, together with the four shift or meta-keys, give a total of 15 possible meta functions that can be invoked:

Key	Normal	Shift	Caps Lock	Ctrl	Alt	Action
Print-Screen/	FN_PRINT	FN_IGNORE	FN_PRINT	FN_LOG	FN_IGNORE	FN_SET
SysRq						
Scroll-Lock	FN_SCROLL	FN_SWAP	FN_SCROLL	FN_SWRESET	FN_SWITCH	FN_CLICK
Pause/Break	FN_DEBUG	FN_DEBUG	FN_DEBUG	FN_IGNORE	FN_KILL	FN_RESET

Keyboard META Functions

FN ACTION

(Usually bound to the < Action > key for the IBM RT/PC, and to the Right Ctrl key for the IBM 6152 Academic System.) Puts the keyboard into action mode. Action mode selects the "Action" set of key values.

FN ALT

(Usually bound to < Alt > .) Puts the keyboard into alt mode. Alt mode selects the "Alt" set of key values.

FN BREAK

Not usually specified by the user; it is invoked when a "make-break" key is released by the user and should only be bound to that single hardware code.

FN_CLICK

(Usually bound to <Action> < Scroll/Lock>.) Causes a different "key click" mode to be selected. There are four such modes: no click, hardware click, software click, and both hardware and software click. Not supported for the IBM 6152 Academic System.

FN_DEBUG

(Usually bound to the < Pause/Break > key.) Invokes the debugger (see debug(8)). Once in console-debugger mode, a "go" command can be used to continue system operation. On systems without the debugger, the screen still displays DEBUG in the status line; but the debugger will not be entered.

The system-attention sequence (<Ctrl>-<Alt>-<Scroll Lock>) can also be used to enter the debugger; use this only if the FN-DEBUG function does not work. On systems without the debugger, the system-attention sequence forces the system to core dump and reboot.

FN IGNORE

(Usually bound to keys that neither generate characters nor activate meta-functions.) Causes that key to be ignored.

FN KILL

(Usually bound to < Alt > < Pause/Break > .) Causes a HANGUP signal (see signal(2)) to be sent to the processes currently in the console terminal process group. It also marks the console terminal as closed. This is sufficient to stop most programs that do not catch or ignore the HANGUP signal. This is analogous to hanging up a dialup line to drop the connection. If the function is invoked again while the console terminal is marked as closed, it generates a KILL signal to all processes in the current process group.

FN NUM LOCK

(Usually bound to < Num-Lock > .) Puts the keyboard into num-lock mode, and activates the standard translation table. The programmable translation table is reactivated when the FN NUM LOCK function is next generated.

FN PRINT

(Usually bound to < Print-Screen > .) Causes the current console screen image to be sent to the line printer.

FN LOG

(Usually bound to < Ctrl>-< Print-Screen>.) Turns hardcopy logging of the screen on and off. Kernel/user logging is independent of debugger logging.

FN_RESET

(Usually bound to < Action > < Pause-Break > .) Causes the keyboard definitions to be reset to their standard values. This is also automatically done when the console terminal is closed, in order to ensure that the next user to log in receives a normal keyboard.

FN_SWAP

(Usually bound to < Shift > < Scroll-Lock > .) Exchanges the bindings of the next two keys typed. This is particularly useful for swapping the definitions of the < Caps-Lock > key and the < Ctrl > key.

FN_SHIFT

(Usually bound to the keys labeled < Shift > on either side of the keyboard.) Puts the keyboard into shifted mode.

FN_SWITCH

(Usually bound to < Alt > < Scroll-Lock > .) Causes the input focus to be switched to the next available display device (see cons(4)).

FN_SWRESET

(Usually bound to < Ctrl > < Scroll-Lock > .) Causes the input focus to be switched to the next available display device, and the display to be re-initialized. If necessary, this involves loading microcode; in all cases, the display is cleared and the cursor moved to the top left corner.

FN SCROLL

(Usually bound to < Scroll-Lock > .) Causes normal output to the console display to be locked out by generating the current XOFF character (see *tty*(4)). Output stops, and the Scroll-Lock light on the keyboard comes on. When the Scroll-Lock light is on, pressing the < Scroll Lock > key again generates the current XON character, which causes output to resume and the Scroll Lock to go off. Note that if the console is in RAW mode, the < Scroll Lock > key only generates XOFF characters.

FN SET

(Usually bound to < Action > < Print-Screen > .) Sets a new value into the keyboard translation table that determines the ASCII characters generated for particular scan codes (see below).

FN_BEEP

(Not usually bound to a key.) Sounds the bell (a beep) when invoked.

FN CAPS LOCK

(Usually bound to < Caps-Lock > or to < Ctrl > .) Puts the keyboard into Caps-Lock mode. The keyboard leaves Caps-Lock mode when the FN_CAPS_LOCK function is next generated. Caps-Lock mode selects the "Caps-Lock" set of key values.

FN CONTROL

(Usually bound to < Ctrl > or to < Caps-Lock > .) Causes the keyboard to go into Control mode. Control mode selects the "Control" set of key values.

NOTES

When the keyboard has more than one mode in effect, the key value selected is that of the highest priority mode. The mode priority ordering is: normal, shift/caps-lock, control, alt, and action (normal being low and action being high).

Using the shift key when the keyboard is in caps-lock mode generates lower case alphabetic characters, and shifts non-alphabetic characters.

When the keyboard is in num-lock mode, i.e the standard translation table is being used, any key-board definition will only be added to the programmable translation table; it does not change the standard set.

The keyboard adapter also supports the speaker, which is activated when the ASCII character bel (^G) is sent to the display. For additional speaker control information, see speaker(4).

It is possible to change the association of keyboard functions and the ASCII strings generated by the keyboard driver in two ways: from the keyboard and from within a program. It is convenient to have a set of definitions (or bindings) done every time a user logs on the system; this is best done with the pf(1) utility invoked from one's *.login* file during login. If this utility is unavailable, or the user wishes to redefine a key temporarily, it may be done from the keyboard. The keyboard procedure is:

- 1. Invoke the FN_SET function (normally bound to <Action > < Print-Screen >). This causes KEY? to be displayed in the status line.
- 2. Enter the keystroke for the key to be set (this includes using the appropriate shift (or meta) key such as < Shift>, < Alt>, < Ctrl>, or < Action>). The status line will show DEF when the keystroke has been accepted.
- 3. Enter the definition. It echoes on the status line as it is accepted. Note that control keys are displayed as ^, followed by the appropriate letter.
- 4. Enter the keystroke for the key being set (as for step 2). This completes the definition.

If during definition you decide to cancel the definition being entered, re-invoke the FN_SET function.

A few ioctl(2) calls apply to the keyboard device, /dev/console, and have the form:

struct kbdarg *arg;

```
#define KBD INDEX NORMAL
                                     0
                                            /* normal position */
#define KBD INDEX SHIFT
                                     1
                                            /* shifted */
#define KBD_INDEX CAPS
                                            /* caps-lock'ed */
                                     2
#define KBD INDEX CONTROL
                                     3
                                            /* control'ed */
#define KBD INDEX ALT
                                     4
                                            /* alt'ed */
                                     5
#define KBD INDEX ACTION
                                            /* action'ed */
/* The following four lines are not supported for the IBM 6152 Academic System */
#define CLICK OFF
                             0x00
#define CLICK HARD
                             0x01
                                            /* hardware click */
#define CLICK SOFT
                                            /* software click */
                             0x02
#define CLICK BOTH
                             0x03
                                            /* both */
#define KBDCGET
                             _IOWR(k,0,struct kbdarg) /* get current def'n */
#define KBDCSET
                              IOW(k,1,struct kbdarg)
                                                           /* set new definition */
#define KBDCRESET
                             _IO(k,2)
                                            /* reset keyboard to standard */
#define KBDCRST
                             IO(k,3)
                                            /* reset (clear) string table */
                                            /* set standard table */
#define KBDCSSTD
                              IO(k,4)
#define KBDSGET
                              IOR(k,5,int)
                                            /* get available string space */
                                            /* get click */
                              IOR(k,6,int)
#define KBDGCLICK
                                            /* set click */
#define KBDSCLICK
                             IOW(k,7,int)
#define KBDSSECURE
                             _IOW(k,8,int)
                                            /* set secure bit and */
                                            /* disable keyboard */
```

The applicable codes are:

is unused.

KBDCRESET

KBDCGET

KBDRST Clear the string definitions. This makes space available for a new set of strings to be defined. It causes any key with a definition of more than two bytes to be set to the FN_IGNORE function.

KBDSSTD Replace the standard definition with the current definition.

KBDSGET Store the number of bytes available for string definitions in the int pointed to by arg.

Store the current definition for scancode kbd scan at index kbd index in the

struct kbdarg pointed to by arg. The keyboard index is one of

KBD INDEX NORMAL, KBD INDEX SHIFT, KBD INDEX CAPS,

Reset the keyboard definitions to the standard definitions. The arg argument

KBD_INDEX_CONTROL, KBD_INDEX_ALT or

KBD_INDEX_ACTION. On the ioctl, the kbd_length contains the maximum length to return; upon return it contains the actual length stored in the

keyboard. The definition is returned into kbd text.

KBDCSET Store a new definition for scancode kbd scan at index kbd index in the key-

> board from struct kbdarg pointed to by arg. The keyboard index is one of KBD INDEX NORMAL, KBD INDEX SHIFT, KBD INDEX CAPS,

KBD INDEX CONTROL, KBD INDEX ALT or

KBD INDEX ACTION. kbd length contains the length of the definition

text in kbd text.

KBDGCLICK Store the current keyboard click status into the int pointed to by arg. The

value is one of CLICK OFF, CLICK HARD, CLICK SOFT or

CLICK BOTH. Not supported for the IBM 6152 Academic System.

KBDSCLICK

Set the current keyboard click status from the *int* pointed to by *arg*. The value must be one of CLICK_OFF, CLICK_HARD, CLICK_SOFT or CLICK_BOTH. Not supported for the IBM 6152 Academic System.

KBDSSECURE

Set arg as the primary boot device to use on reboot, and disable the keyboard. Definitions for arg may be found in the include file < machine/nvram.h > and have the legal values of NVR_FD0, NVR_FD1, NVR_FD2, NVR_FD3, NVR_HD0, NVR_HD1, NVR_HD2, and NVR_HD3. If arg is zero and the console is secured, KBDSSECURE reenables the keyboard and resets the boot order to its default. The effects of this ioctl may also be removed by locking and then unlocking the keylock on the front panel. Not supported for the IBM 6152 Academic System.

ERRORS

The following errors can be returned by the driver:

[EINVAL] The kbd scan or kbd index is invalid.

[E2BIG] The string being defined exceeds the available space.

[EPERM] The caller is not the super-user (for the KBDSSTD ioctl).

FILES

For the IBM RT PC:

/dev/console

/dev/ttyaed

/dev/ttyap16

/dev/ttyap8c

/dev/ttyapa8

/dev/ttyega

/dev/ityega

/dev/ttymono

/dev/ttympel

/dev/aed

/dev/apa16

/dev/apa8c

/dev/apa8

/dev/ega

/dev/mono

/dev/mpel

For the IBM 6152 Academic System:

/dev/tty8514

/dev/ttyvga

/dev/vga

/dev/ibm8514

SEE ALSO

pf(1), cons(4), ibm5081(4), ibm5151(4), ibm6153(4), ibm6154(4), ibm6155(4), ibm8514(4), ibmaed(4), speaker(4), tty(4), vga(4), xemul(4), debug(8), reboot(8)

DIAGNOSTICS

None.

NAME

lan - IBM RT PC Token-Ring Adapter

SYNOPSIS

For the IBM RT PC:

controller lanc0 at iocc0 csr 0xf00001c0 priority 12 device lan0 at lanc0 drive 0

For the IBM 6152 Academic System:

device lan0 at iocc0 csr 0x00000a20 priority 3 device lan1 at iocc0 csr 0x00000a24 priority 10

DESCRIPTION

The lan interface provides access to the IBM local area network, a token-ring, star-wired network accommodating up to 260 attaching devices.

On the IBM RT PC, the hardware has 16k bytes of RAM, of which 4k bytes are used for transmit buffers and 12k for receive buffers. Simultaneous transmission and reception of data are supported. The adapter operates as a DMA bus master. The use of trailers is negotiated with ARP. This negotiation may be disabled on a per-interface basis by setting the IFF_NOTRAILERS flag with an SIOCSIFFLAGS ioctl.

The host's Internet address is specified at boot time with an SIOCSIFADDR ioctl. The *lan* interface uses the address resolution protocol described in *arp*(4P) to map dynamically between Internet and token-ring addresses on the local network.

On the IBM 6152 Academic System, the interface has 8K bytes of expansion RAM, of which 3K bytes are used for transmit buffers and 5K for receive buffers. Simultaneous transmission and reception of data are supported.

The lan interface uses the address resolution protocol, described in arp(4P), to map dynamically between Internet and token-ring addresses on the local network.

The lan driver uses the new token ring packet format (the bridge and snap options on) as the default configuration.

DIAGNOSTICS

For the IBM RT PC:

lan%d: address xx:xx:xx:xx:xx:xx The hardware address used to open the adapter is displayed.

lan%d: token ring adapter initialization failure, status = 0x%b, error code = 0x%x. An unrecoverable hardware error has occurred during initialization of the adapter.

If status indicates TEST and ERR, the bring-up diagnostic error codes are:

Code	Description
XX30 or XXB0	Initial test error
XX31 or XXB1	ROM CRC error
XX32 or XXB2	RAM error
XX33 or XXB3	Instruction test error
XX34 or XXB4	Context/interrupt test error
XX35 or XXB5	Protocol handler hardware error
XX36 or XXB6	System interface register error

If status indicates INIT and ERR, the initialization error codes are:

LAN(4)

Code	Description
XX58 or XXD8	DIO parity error
XX59 or XXD9	DMA timeout
XX5A or XXDA	DMA parity error
XX5B or XXDB	DMA bus error
XX5C or XXDC	DMA data error
XX5D or XXDD	Adapter check

lan%d: dma channel %x. The adapter has not gained exclusive use of the necessary system dma channel. The adapter is unusable.

lan%d: unrecoverable token ring adapter failure, interrupt code = 0x%x, parm0 = 0x%x, parm1 = 0x%x, parm2 = 0x%x. An unrecoverable adapter error has occurred.

Code	Parm0	Parm1	Parm2	Description
0001	*	*	*	Program Check
0002	R13	R14	R15	Invalid XOP
0004	R13	R14	R15	Invalid Error Interrupt
8000	R13	R14	R15	Invalid Interrupt
0010	XXXX	XXXX	XXXX	Ring Overrun
0020	XXXX	XXXX	XXXX	Ring Underrun
0040	R13	R14	R15	XMIT Parity Error
0800	R13	R14	R15	RECV Parity Error
0100	R13	R14	R15	PH Parity Error
0200	R13	R14	R15	RIF Parity Error
0400	R13	R14	R15	EM Parity Error
0800	R13	R14	R15	LB Parity Error
1000	R13	R14	R15	Illegal Op Code
2000	0000	Sys Address	Sys Address	DMA Write Abort - timeout
2000	0001	Sys Address	Sys Address	DMA Write Abort - parity error
2000	0002	Sys Address	Sys Address	DMA Write Abort - bus error
4000	0000	Sys Address	Sys Address	DMA Read Abort - timeout
4000	0001	Sys Address	Sys Address	DMA Read Abort - parity error
4000	0002	Sys Address	Sys Address	DMA Read Abort - bus error
8000	XXXX	XXXX	XXXX	DIO Parity Error

^{* --} is an abend code containing information about the program check.

lan%d: single station on network. The adapter can transmit/receive frames only to/from itself. This state is normal for the first station on the ring; however, it also occurs when the adapter is not inserted on the ring.

lan%d: hardware error, adapter removed from ring.

lan%d: beaconing. A break has occurred in the ring between this station and its nearest addressable upstream neighbor. The ring is in an automatic process of being restored.

lan%d: cable failure. The cable connecting the adapter to the ring is either open or shorted. The driver will attempt to reopen the adapter after 60 seconds.

lan%d: removed from network. A remove MAC (Media Access Control) frame was received from an authorized LAN Manager. The adapter is closed and removed from the ring.

lan%d: node address error. An error was detected in the address used in opening the adapter. The adapter remains closed.

lan%d: open err = 0x%x, function failure. Hardware failure detected during open processing. This may be due to a cable problem. The driver will attempt to reopen the adapter up to 3 times.

lan%d: open err = 0x%x, receiver exception. A loss of signal has been detected at receiver input. The driver will attempt to reopen the adapter after 60 seconds.

lan%d: open err = 0x%x, ring failure. The driver will attempt to reopen the adapter after 60 seconds.

lan%d: open err = 0x%x, ring beaconing. The driver will attempt to reopen the adapter after 60 seconds.

lan%d: open err = 0x%x, duplicate node address. Two stations on the ring have the same physical address. The adapter remains closed.

lan%d: open err = 0x%x, timeout. The adapter failed to insert onto the ring within 18 seconds. The driver will attempt to reopen the adapter up to 3 times.

lan%d: open err = 0x%x, request parameters. The ring parameter server failed to respond to a request-initialization MAC (Media Access Control) frame. The driver will attempt to reopen the adapter up to 3 times.

lan%d: open err = 0x%x, remove received. The adapter remains closed.

lan%d: command reject, internal software error, reject reason = 0x%x.

lan%d: adapter jammed. The adapter has not cleared the system control block within 5 seconds. The state of the adapter is unknown.

lan%d: can't handle af%d. The interface was handed a message with address formatted in a unsuitable address family; the packet was dropped.

For the IBM 6152 Academic System:

lan%d: adapter check. The adapter has encountered an unrecoverable error and is detached from the ring.

lan%d: adapter error. An unexpected adapter timer interrupt, adapter machine check, adapter deadman timer interrupt, 40-millisecond timer overrun, shared RAM access violation or an illegal operation to an attachment control area register has occurred.

lan%d: ARB new ring status = 0x%x. The adapter is reporting a new ring status. See "Ring Status" in the "Technical Reference Token-Ring Network PC Adapter" IBM Personal Computer Hardware Reference Library.

lan%d: can't handle af%d. The interface was handed a message with addresses formatted in an unsuitable address family; the packet was dropped.

SEE ALSO

intro(4N), inet(4F), arp(4P)

LP(4)

NAME

lp - line printer

SYNOPSIS

For the IBM RT PC:

device lp0 at iocc0 csr 0xf00003bc priority 7 device lp1 at iocc0 csr 0xf0000378 priority 7 device lp2 at iocc0 csr 0xf0000278 priority 5

For the IBM 6152 Academic System: device lp0 at iocc0 csr 0x000003bc priority 7

DESCRIPTION

Lp provides the interface to the IBM 5152 Graphics Printer and the IBM 4201 Proprinter using the parallel port of the monochrome adapter or PC/AT serial/parallel adapter (for the IBM RT PC), or the plnar parallel port of the PS/2 Model 60. Bytes are sent to the printer as is; the driver does no interpretation. No automatic page-ejects are generated; this can be done by the print spooling system. See "Line Printer Spooler Manual" in 4.3BSD UNIX System Manager's Manual.

The IBM 4201 Proprinter switch settings should be as follows:

switches 3,4,7 off switch 6 on

switch 1,2,5

optional - See IBM 4201 Proprinter Guide to Operations,

PN6328945, for effects of these switches

Although a priority must be specified in a configuration entry, the driver does not currently use interrupts on the IBM 6152 Academic System.

The unit number of the printer is specified by the minor device divided by 8. Currently, the low three bits are ignored.

FILES

/dev/lp[012]

SEE ALSO

lpr(1)

Chapter 6 of the IBM 4201 Proprinter Guide to Operations, PN6328945

DIAGNOSTICS

lp%d: offline.

lp%d: out of paper.

lp%d: unknown printer error, status = 0x%b. The printer did not become ready in 20 seconds, for some reason other than being off-line or out of paper.

NAME

mem, kmem, kmem1, kmem2, kmem4, ros, afpamem - main memory

DESCRIPTION

Mem is a special file that is an image of the main memory of the computer. It may be used, for example, to examine -- and even to patch -- the system.

Byte addresses in *mem* are interpreted as physical memory addresses. References to non-existent locations cause errors to be returned. The I/O space (segment 0xf) cannot be accessed through this special file.

The file *kmem* is the same as *mem* except that kernel virtual memory, rather than physical memory, is accessed. Although I/O space can be accessed through this file, this use of *kmem* is discouraged, since the width of the access (byte, halfword or fullword) is not controllable by the user.

The special file /dev/bus (bus(4)) can be used to gain direct access to I/O and memory addresses on the system I/O bus.

For the IBM RT PC only:

The files kmem1, kmem2, and kmem4 are similar to kmem except that they guarantee access by bytes, halfwords and fullwords, respectively. They are provided primarily to allow access to the I/O space of the IBM RT PC (segment 0xf), although any valid kernel virtual address may be given. When using a kmemx file, both the user's buffer and the file offset must be aligned on an x-byte boundary, and the length must be a multiple of x bytes. If these conditions are not met, both read(2) and write(2) will fail with errno set to EINVAL.

Examining and patching device registers is likely to lead to unexpected results when read-only or write-only bits are present.

On the IBM RT PC, the I/O space begins at location 0xf0000000 of kmem, kmem1, kmem2, and kmem4. The current implementation places per-process data for the current process at virtual address 0x1ffff87c. Since this is subject to change, the use of nlist(3) by programs (and nm(1) at the command level) to obtain the value of u from vmunix is strongly recommended.

The file ros is similar to mem, except that it refers to the Read Only Storage on the processor card. There is no ros on the IBM 6152 Academic System.

Afpanem is used to check, load, and verify the control store of the Advanced Floating Point Adapter (AFPA). This control store consists of 64-bit words; data must be read or written in multiples of 64 bits (eight bytes; two words). The data must be written to, or read from, an address in the user's address space which is word aligned. Control store access to the AFPA must be enabled before afpanem access is allowed. Note that the normal user of afpanem is afpacode(8R). The AFPA is valid only on the IBM RT PC.

FILES

/dev/mem

/dev/kmem

/dev/kmem1

/dev/kmem2

/dev/kmem4

For the IBM RT PC only:

/dev/ros

/dev/afpamem

SEE ALSO

bus(4), afpacode(8R)

MEM(4) MEM(4)

BUGS

There is no driver to do ior's and iow's to the CPU's I/O bus. Thus it is not possible to access the registers of the memory management unit and the I/O interface modules.

If more than 8M of physical address space is used by memory and DMA is in use, reads of /dev/ros will fail.

MOUSE(4) MOUSE(4)

NAME

mouse - mouse interface

SYNOPSIS

pseudo-device ms

DESCRIPTION

The mouse driver provides a low-level interface to the mouse for the IBM RT PC or IBM 6152 Academic System. A tty(4)-like interface is provided, in that the kernel uses internal tty protocols for buffering data to read data from the interface. Thus, the system calls select(2) and fcntl(2) relating to non-blocking I/O may be used. Commands are issued to the mouse through the standard ioctl(2) mechanism, although none of the terminal-specific ioctls apply.

Data

The data structures necessary for communicating with the mouse are provided in < machineio/mouseio.h>. This file also includes macros for interpreting the mouse stream data, and symbolic names for the commands the mouse understands. This is only valid if the mouse line discipline MSLINEDISC (defined in < machineio/mouseio.h >) is used.

Since many applications expect a three-button mouse, a three-button mouse is simulated by indicating a middle button when both buttons are depressed. This means that an application program will never see a data report indicating that both buttons are pressed. There is a small delay in recognizing a single button, in order to allow smooth recognition of both buttons, since one cannot always press both buttons at exactly the same time. In addition, a single button from a middle (double-button) transition will not be reported until both buttons have been released and then the single button pressed.

Commands

The commands available to the user through ioctls are defined in < machineio/mouseio.h > and described below. When in stream mode, the driver will disable the mouse before issuing commands which expect responses.

MSIC STREAM

Sets the mouse in stream mode which, when enabled, sends unsolicited data reports. This is the default after the mouse has been opened.

MSIC REMOTE

Sets the mouse to remote mode, where it will send data reports only in response to a MSIC READXY command; no unsolicited data will be sent. Not supported for IBM 6152 Academic System.

MSIC STATUS

The current mouse status data report, as defined in < machineio/mouseio.h >, will be placed in the passed four-byte character pointer. Note that the status report contains the true state of the buttons; a middle button is not simulated.

MSIC READXY

This will solicit a data report from the mouse even if it hasn't been moved or button conditions have not changed. MSIC READXY can be used in either stream or remote mode. In stream mode, the driver will disable the mouse before performing the read. Not supported for IBM 6152 Academic System.

MSIC_ENABLE

Enable the mouse to send unsolicited data reports in stream mode. This is the default after the mouse has been opened.

MSIC DISABLE

Prohibits the mouse from sending unsolicited data reports in *stream* mode.

MSIC_EXP

For the IBM RT/PC, an exponential transform is applied to data reported by the mouse, as follows:

Let x be the number of counts accumulated in the sample interval. The value reported will be $sign(x) \cdot (2^N)$ where

$$N = \frac{\text{abs}(x) \cdot \text{rate}}{\text{resolution}} - 3$$

This scaling applies only to mouse-generated reports. Reports generated in response to MSIC READXY are always linear.

For the IBM 6152 Academic System, MSCI_EXP exchanges the scaling factor applied to mouse-generated reports from linear (a:T) to 2:1:

Let x be the number of counts accumulated in the sample interval. The value reported will be 2x.

MSIC LINEAR

Set the mouse back to linear scaling, where the value reported is the number of counts. This is the default after the mouse has been opened.

MSIC_SAMP

Set the mouse sampling rate. The argument passed in the *ioctl* must contain the desired sampling reate in counts per second. Legal values are 10, 20, 40, 60, 80 and 100. These are provided in < machineio/mouseio.h > as MS_RATE_10, MS_RATE_20, ..., MS_RATE_100 respectively. The default after the mouse has been opened is MS_RATE_100.

MSIC RESL

Set the mouse resolution. The argument passed must contain the the required resolution in counts per inch. Legal resolutions are 25, 50, 100, 200. These are provided in < machineio/mouseio.h > as MS_RES_25, MS_RES_50, MS_RES_100, MS_RES_200, respectively. The default after the mouse has been opened is MS_RES_100.

The mouse data stream follows to console input focus (see cons(4)). There are seven mouse devices, one generic mouse (/dev/mouse) and one for each supported display (/dev/msxxx). If the generic mouse is opened, it takes input from the current input focus. The last four bits of the minor device decide which mouse data stream the process reads (0 = generic mouse). The upper four bits select the default line discipline used by the mouse.

FILES

For the IBM RT/PC:

/dev/mouse

/dev/msaed

/dev/msapa16

/dev/msapa8c

/dev/msapa8

/dev/msega

/dev/msmono

For the IBM 6152 Academic System:

/dev/mouse

SEE ALSO

select(2), tty(4), tb(4)

MOUSE(4)

For the IBM RT/PC: IBM RT PC Hardware Technical Reference Manual, "Keyboard, Locator, Speaker Adapter", section 5, SV21-8021

MTIO(4)

NAME

mtio - 4.3/RT magtape interface

DESCRIPTION

The files st0, rst0, nst0, nrst0 refer to the IBM RT PC streaming tape drive, st(4). The files st0 and rst0 are rewound when closed; the others are not. When a file open for writing is closed, two ends-of-file are written.

A standard tape consists of a series of 512 byte records terminated by an end-of-file. To the extent possible, the system makes it possible, if inefficient, to treat the tape like any other file. Seeks have their usual meaning and it is possible to read or write a byte at a time. Due to hardware limitations, reads and writes cannot be intermingled, and writing always truncates the tape (not just the file) at that point.

The st files discussed above are useful when it is desired to access the tape in a way compatible with ordinary files. When foreign tapes are to be dealt with the "raw" interface is appropriate. The associated files are named rst0 and nrst0. A number of other ioctl operations are available on raw magnetic tape. The following definitions are from $\langle sys/mtio.h \rangle$:

```
* Structures and definitions for mag tape io control commands
/* structure for MTIOCTOP - mag tape op command */
struct mtop
        short mt op;
                                       /* operations defined below */
                                       /* how many of them */
        daddr t mt count;
};
/* operations */
#define MTWEOF
                               0
                                       /* write an end-of-file record */
#define MTFSF
                               1
                                       /* forward space file */
#define MTBSF
                               2
                                       /* backward space file */
                               3
#define MTFSR
                                       /* forward space record */
                               4
#define MTBSR
                                       /* backward space record */
                               5
#define MTREW
                                       /* rewind */
                               6
                                       /* rewind and put the drive offline */
#define MTOFFL
                               7
                                       /* no operation, sets status only */
#define MTNOP
#define MTERASE
                               8
                                       /* for streamer: erase tape */
#define MTRETENSION
                                       /* for streamer: retension tape */
/* structure for MTIOCGET - mag tape get status command */
struct
       mtget
              -{
                                       /* type of magtape device */
        short
              mt type;
/* the following two registers are grossly device dependent */
        short
               mt dsreg;
                                       /* "drive status" register */
               mt_erreg;
                                       /* "error" register */
        short
/* end device-dependent registers */
        short
               mt resid;
                                       /* residual count */
/* the following two are not yet implemented */
                                       /* file number of current position */
        daddr t mt fileno;
        daddr t mt blkno;
                                       /* block number of current position */
/* end not yet implemented */
};
```

MTIO(4)

```
* Constants for mt type byte
#define MT ISTS
                            0x01
#define MT ISHT
                            0x02
#define MT ISTM
                            0x03
#define MT ISMT
                            0x04
#define MT ISUT
                            0x05
#define MT_ISCPC
                            0x06
#define MT ISAR
                            0x07
#define MT ISST
                            0x08
/* mag tape io control commands */
#define MTIOCTOP
                     IOW(m, 1, struct mtop)
                                                  /* do a mag tape op */
#define MTIOCGET
                     IOR(m, 2, struct mtget)
                                                  /* get tape status */
#ifndef KERNEL
#define DEFTAPE
                     "/dev/rmt12"
#endif
```

Each read or write call reads or writes the next record on the tape. In the write case the record has the same length as the buffer given. During a read, the record size is passed back as the number of bytes read, provided it is no greater than the buffer size; if the record is long, an error is indicated. In raw tape I/O seeks are ignored. A zero byte count is returned when a tape mark is read, but another read will fetch the first record of the new tape file.

Note that certain tapes do not have this capability. See the device manual page for details.

FILES

```
dev/{n,}st
dev/{n,}rst
```

SEE ALSO

mt(1), tar(1), tp(1), st(4)

BUGS

The status should be returned in a device-independent format.

The default tape doesn't apply to the IBM RT PC.

PSP(4)

NAME

psp - planar serial port RS232C interface

SYNOPSIS

device psp0 at iocc0 csr 0xf0008000 flags 0x03 priority 2

DESCRIPTION

The floor-model IBM RT PC comes with two planar serial ports with full modem control; the desk model does not. Each port behaves as described in tty(4) and may be set to run at any of 16 speeds; see tty(4) for the encoding.

Bit i of flags may be set to 1 to specify that the ith port should be treated as having carrier initially present. For example, specifying "flags 0x01" in the configuration entry for psp0 causes only the line ttys0 to be treated in this way.

Note that \(\frac{dev}{ttys} \) corresponds to the port labeled "S1" and \(\frac{dev}{ttys} \) corresponds to the port labeled "S2" on the back of the IBM RT PC.

NOTES

Psp is not supported on the IBM 6152 Academic System.

FILES

/dev/ttys[01]

SEE ALSO

tty(4)

DIAGNOSTICS

psp%d: overrun error. Data has been lost because characters are arriving at the adapter faster than the driver can read them.

RVD(4P) RVD(4P)

NAME

rvd - Remote Virtual Disk protocol

DESCRIPTION

RVD is a network service which allows several physical machines to share one physical mass storage device such as a hard disk. The basic concept is to have the machine to which the device is physically attached act as a server to read and write blocks for all the other machines desiring use of the resource.

The server program apportions the physical blocks into "virtual disk packs" based on a table maintained with vddb(8). The packs can then be used separately by clients. There are three modes of use: read-only, shared, and exclusive. Exclusive mode is used for read-write access, while read-only mode is as it sounds. Shared mode is not supported under 4.3/RT. If a disk pack is "spun up" in read-only mode, several clients may share the pack and read its information. In exclusive mode, one client has exclusive use of the disk pack.

Packs are "spun up" and "spun down" with the up and down commands (see up(1)). This can be done at reboot time within /etc/rc.local (see rc(8)) or at login time within $^{-}/.login$ (see csh(1)). Once a pack is spun up, it behaves like a disk physically attached to the local machine (excepting network latency). The client can do anything desired with the pack; both MS-DOS and UNIX operating system file systems have been used on the same physical drive at the same time (on separate packs, of course).

RVD is implemented in two parts: server code and client code. The server code is written as a user process, i.e. it does not require any special privileges beyond read/write access to the disks it manages. The server opens a network socket and listens for UDP connections. It also accepts all RVD packets and acts on them. RVD is a protocol different from both UDP and TCP, although similar in nature to the former.

The client code is implemented as a pseudo-device and corresponding device driver in the kernel. It can handle up to 10 remote virtual disks simultaneously, which are associated with the pseudo-devices below.

FILES

/dev/vd[0-9]a block special file pseudo-device /dev/rvd[0-9]a character special file pseudo-device

SEE ALSO

up(1), rvddb(5), rvdtab(5), rvdflush(8), rvdchlog(8), rvddown(8), rvdexch(8), rvdflush(8), rvdslog(8), rvdsend(8), rvdshow(8), rvdshow(8), rvdsrv(8), savervd(8), spinup(8), vddb(8), vdstats(8) "The Remote Virtual Disk System" in Volume II, Supplementary Documents

SC(4)

NAME

sc - IBM 9332 disks using the IBM Small Computer System Interface (SCSI) Adapter

SYNOPSIS

controller scc0 at iocc0 csr 0xf0000d52 priority 11 controller scc1 at iocc0 csr 0xf0000952 priority 12 disk sc0 at scc0 drive 0 disk sc1 at scc0 drive 1 disk sc2 at scc0 drive 2 disk sc3 at scc0 drive 3 disk sc4 at scc0 drive 4 disk sc5 at scc0 drive 5 disk sc6 at scc0 drive 6 disk sc7 at scc1 drive 0 disk sc8 at scc1 drive 1 disk sc9 at scc1 drive 2 disk sc10 at scc1 drive 3 disk sc11 at scc1 drive 4 disk sc12 at scc1 drive 5 disk sc13 at scc1 drive 6

DESCRIPTION

This driver supports the IBM 9332 disk unit, models 200, 220, 400, and 440 connected to an IBM Small Computer System Interface (SCSI) Adapter. The driver can support up to two adapter cards, each driving up to seven IBM 9332 disk units.

Files with minor device numbers 0 through 6 refer to partitions of drive 0; minor devices 7 through 13 refer to drive 1, and so on. The standard device names begin with "sc", followed by the drive number and then a letter a-h for partitions 0-7, respectively. The drive number is the setting of the drive address switch in the back of the disk unit, plus 7 if the disk unit is connected to the second adapter. (See *IBM 9332 Disk Unit, Installing*, SA21-9804 for more information.)

The block files access the disk via the system's normal buffering mechanism and may be read and written without regard to physical disk records. There is also a "raw" interface that provides for direct transmission between the disk and the user's read or write buffer. A single read or write call results in exactly one I/O operation; therefore, raw I/O is more efficient when many words are transmitted. The names of the raw files conventionally begin with an extra r.

In raw I/O, counts should be multiples of 512 bytes (a disk sector). Likewise, seek(2) calls should specify multiples of 512 bytes. The user's buffer should be aligned on a fullword boundary when the raw device is used.

DISK SUPPORT

The origin and size (in sectors) of the default partitions on each drive are as follows. The character "?" stands for a hex drive number in the range 0-d.

SCSI 200 200 Mbyte SCSI disk unit partitions (Models 240, 260)

disk	start	length	capacity	cylinders	
sc?a	1	15884	7.8 MB	1 - 54	
sc?b	55	33440	16.3 MB	55 - 167	
sc?c	0	391016	190.0 MB	0 - 1014	
sc?d	168	15884	7.8 MB	168 - 221	
sc?e	222	55936	27.3 MB	222 - 410	
sc?f	411	267880	130.8 MB	411 - 1315	
sc?g	168	339808	165.9 MB	168 - 1315	
sc?h		Not Used			

SCSI 400 400 Mbyte SCSI disk unit partitions (Models 440, 460)

disk	start	length	capacity	cylinders	
sc?a	1	15884	7.8 MB	1 - 54	
sc?b	55	66880	32.6 MB	55 - 280	
sc?c	0	782328	381.9 MB	0 - 2642	
sc?d	1266	15884	7.8 MB	1266 - 1319	
sc?e	1320	307200	150.0 MB	1320 - 2357	
sc?f	2358	82880	40.5 MB	2358 - 2637	
sc?g	1266	406112	198.3 MB	1266 - 2637	
sc?h	281	291346	142.3 MB	281 - 1265	

It is unwise for all these special files to be present in one installation, because addresses overlap and protection becomes a sticky matter. The sc?a partition can be used for the root file system, the sc?b partition as a paging area, and the sc?c partition for access to the entire disk, including boot and configuration information at the start of the disk and bad-block and diagnostic regions at the end of the disk. The sc?c partition may be used as a filesystem. Standard partition tables are calculated using the diskpart(8) program. Non-standard disk partitions may be created using the minidisk(8R) utilities.

FILES

/dev/sc[0-9a-d][a-h]	block files
/dev/rsci0-9a-dlla-hl	raw files

SEE ALSO

diskpart(8), minidisk(8R), newfs(8)

DIAGNOSTICS

SC%d: unexpected condition tag status. The hardware returned an interrupt for a transfer that has not been started.

SC%d: %s condition status (%r) %s. The disk unit returned an error condition code. The types, printed by the first %s, are CHECK, BUSY, INTER, RESCFT, and unknown. %x is the status bits returned by the disk unit, and the last %s is either READ or WRITE for the type of operation.

SC%d: last int tag (%d) (%d) The sc detected no interrupt for the operation attached to "tag" after an extended period of time.

OH NO SENSE command didn't work (%)!!! Sc tried to do a sense data command to try to recover from an error, but the sense command failed.

During auto configuration, one of the following messages may appear on the console in recognition of the drive type:

```
sc %d: drive type (440) scsi 400
sc %d: drive type (460) scsi 400
sc %d: drive type (240) scsi 200
sc %d: drive type (260) scsi 200
sc %d: drive type (%s) unknown using scsi 200
```

After the auto configuration message, the following line is printed:

Vid Pid Model RLID ROSPL RAMPL (%s)

Vid is the vendor ID of the sc device attached (should be IBM). Pid is the product ID, 9332. Model is the model number (440, 460, 240, 260). RLID is the ros level (read-only storage). ROSPL is the ros patch level. RAMPL is the ram patch level (random access memory).

SC(4)

BUGS

In raw I/O read(2) and write(2) truncate file offsets to 512-byte block boundaries, and write scribbles zeroes on the tail of incomplete blocks. Thus, in programs likely to access raw devices, read, write, and lseek(2) should always deal in 512-byte multiples.

In raw I/O, the buffer must be aligned on a fullword boundary.

There is no standalone sc driver.

You cannot boot from the sc.

You cannot dump a vmcore to the sc.

SPEAKER(4) SPEAKER(4)

NAME

speaker - console speaker interface

DESCRIPTION

The speaker driver provides a write-only interface to the console speaker. There are no ioctls. The speaker is controlled by multiple writes of struct spk_blk defined in < machineio|speakerio.h>. Writes of less than sizeof(struct spk blk) are not accepted.

The spk blk structure is defined below:

```
/* spk_blk is how a user level program passes a note to the speaker */
struct spk_blk {
    unsigned char volume;
    unsigned char freqhigh;
    unsigned char freqlow;
    unsigned short duration;
};
```

Values written to |dev|speaker are checked according to the following rules:

- 1) volume must be 0,1,2, or 3.
- 2) frequish must be a non-negative integer power of two (i.e. a single bit) ≤ 64 .
- 3) freqlow must be ≥ 19 if freqhigh = 1, otherwise freqlow ≥ 120 (note all characters are unsigned.)
- 4) duration must be < 0x8000.

If the values written do not conform to the above rules, the values are changed to the nearest legal value before processing (for example, writing a volume of 10 gets changed to a volume of 3; writing a freqhigh of 18 gets changed to a freqhigh of 16).

Units for the values are as follows:

- a) Volume: 0 is off, 1 is low, 2 is medium, 3 is high. Only 0 for off and non-zero for on are recognized by the IBM 6152 Academic System.
- b) Freqhigh/Freqlow: To convert a frequency in hertz to freqhigh and freqlow, use the following algorithm:

```
if (freq < 23) {
            b.freqhigh = 0;
            b.freqlow = SPKOLOMIN;
} else if (freq < 46) {
            b.freqhigh = 64;
            b.freqlow = (char) ((6000.0 /(float) freq) - 9.31);
} else if (freq < 91) {
            b.freqhigh = 32;
            b.freqlow = (char) ((12000.0 /(float) freq) - 9.37);
} else if (freq < 182) {
            b.freqhigh = 16;
            b.freqlow = (char) ((24000.0 /(float) freq) - 9.48);
} else if (freq < 363) {</pre>
```

SPEAKER(4) SPEAKER(4)

```
b.freqhigh = 8;

b.freqlow = (char) ((48000.0 /(float) freq) - 9.71);

} else if (freq < 725) {

b.freqhigh = 4;

b.freqlow = (char) ((96000.0 /(float) freq) - 10.18);

} else if (freq < 1433) {

b.freqhigh = 2;

b.freqlow = (char) ((192000.0 /(float) freq) - 11.10);

} else if (freq < 12020) {

b.freqhigh = 1;

b.freqlow = (char) ((384000.0 /(float) freq) - 12.95);

} else {

b.freqhigh = 0;

b.freqlow = SPKOLOMIN;

}
```

c) Duration is in 1/128ths of a second.

The IBM 6152 Academic System rounds this to a granularity of 1/18.5ths of a second.

FILES

/dev/speaker

BUGS

It is possible to queue 20 minutes of sound (5 notes with duration > 4 minutes) to the speaker which cannot be stopped (short of resetting the system).

Because of blocking requirements, output cannot generally be redirected to |dev| speaker. It must be sent to a program, such as dd(1), that will issue writes in multiples of size of (struct spk_blk).

Control over volume is currently unimplemented on the IBM 6152 Academic System.

NAME

st - streaming-tape interface

SYNOPSIS

controller stc0 at iocc0 csr 0xf00001e8 priority 12 tape st0 at stc0 drive 0

DESCRIPTION

The streaming-tape driver provides a QIC 02 (Quarter-Inch Cartridge) tape interface. The device is used to back up data on hard disks. The adapter uses programmed I/O for the 16-bit data transfers to and from its 512-byte sector buffer.

When writing to the tape, the block interface generally gives better performance than the raw interface for transfers smaller than 100 sectors. Note, however, that errors detected when writing to the block interface might not be returned to the program; errors detected when writing to the raw interface are returned.

Writes to the block tape device must be multiples of 2k in length.

When reading the tape, the raw interface gives better performance.

For example, to dump a filesystem g, type:

dump 0f /dev/rst0 /dev/rhd0g

To restore the filesystme to the current directory, type:

```
mount /dev/hd0g /mnt
cd /mnt
restore rvf /dev/rst0
```

Erase brand-new tapes before use, with the command:

mt -f /dev/rst0 erase

FILES

```
/dev/nst0
/dev/nrst0
```

SEE ALSO

```
mt(1), tar(1), mtio(4), dump(8), restore(8)
```

DIAGNOSTICS

st%d: write protected. An attempt was made to write on the tape drive while the tape was write-protected.

st%d: hard error bn%d er = %b. A tape error occurred at block bn. Any error is fatal on non-raw tape; when possible, the driver will retry the failed operation several times before reporting the error.

st%d: lost interrupt. A tape operation did not complete within a reasonable time. This can occur if you open the tape door before the tape drive has been closed or stops moving.

BUGS

The streaming tape hardware backspace operation is exceedingly slow. For this reason the backspace-file command has not been implemented.

STDEMUL(4) STDEMUL(4)

NAME

FILES

stdemul - standard output emulator

DESCRIPTION

The standard output emulator outputs raw characters to the display without interpreting special characters (except bel (^G)). This emulator is used for smart displays which do their own emulation (as does the IBM Academic Information Systems experimental display). To select stdemul when it is not a default, use the following C code:

```
#include < stdio.h >
               #include < machinecons/screen config.h >
                  int fd, output emulator;
                  char *console; /* set to some console device*/
                  fd = open (console, O RDWR);
                  output emulator = E STDOUTPUT;
                  ioctl (fd,EOSETD,&output_emulator);
       /dev/console
       /dev/ttyaed
SEE ALSO
       ibmaed(4), ibmemul(4)
```

"IBM/4.3 Console Emulators" in Volume II, Supplementary Documents

TB(4)

```
NAME
```

tb - line discipline for digitizing devices

SYNOPSIS

pseudo-device tb

DESCRIPTION

This line discipline provides a polled interface to many common digitizing devices (e.g. mouse and tablet) connected to a host. When these devices stream data at high speed, the use of the line discipline is critical in minimizing the number of samples that would otherwise be lost due to buffer exhaustion in the tty(4) handler.

The line discipline is enabled by a sequence:

```
#include < sys/ioctl.h >
#include < sys/tbioctl.h >
int ldisc = TABLDISC, fildes; ...
ioctl(fildes, TIOCSETD, &ldisc);
```

A typical application program then polls the digitizing device by reading a binary data structure which contains: the current X and Y positions (in the device coordinate space), up-down status of the buttons or pen stylus, proximity information (when available), and a count of the number of samples received from the input device since it was opened. Refer to the include file for a complete description.

The line discipline supports *ioctl*(2) requests to get/set the operating mode, and to get/set the tablet type and operating mode by *or*-ing the two values together.

The mouse format returned by tb(4) is:

```
struct tbmspos {
                              /* delta x, positive to the right */
               int
               int
                             /* delta y, positive up */
                      ypos;
                      status; /* button status */
               short
       #define MIDDLE BUTTON
                                     0x2
       #define RIGHT_BUTTON
                                     0x4
       #define LEFT BUTTON
                                     0x0
       #define NO BUTTON
                                     0xOV
                      scount; /* running count of the number of events reported*/
SEE ALSO
       tty(4)
DIAGNOSTICS
       None.
```

NAME

tty - general terminal interface

SYNOPSIS

#include < sgtty.h >

DESCRIPTION

This section describes both a particular special file |dev|tty and the terminal drivers used for conversational computing.

Line Disciplines:

The system provides different *line disciplines* for controlling communications lines. In this version of the system there are five disciplines available for use with terminals:

old The old (Version 7) terminal driver. This is sometimes used when using the standard shell sh(1).

new The standard Berkeley terminal driver, with features for job control; this must be used when using csh(1).

A line discipline used to communicate with the IBM 3812 Pageprinter in asynchronous data mode. It is described in ap(4).

tb,mouse

Line disciplines used for the mouse. See mouse(4), tb(4).

Line discipline switching is accomplished with the TIOCSETD ioctl:

```
int ldisc = LDISC;
ioctl(f, TIOCSETD, &ldisc);
```

where LDISC is OTTYDISC for the standard tty driver and NTTYDISC for the "new" driver. The standard (currently old) tty driver is discipline 0 by convention. Other disciplines may exist for special purposes, such as use of communications lines for network connections. The current line discipline can be obtained with the TIOCGETD *ioctl*. Pending input is discarded when the line discipline is changed.

All of the low-speed asynchronous communications ports can use any of the available line disciplines, no matter what hardware is involved. The remainder of this section discusses the "old" and "new" disciplines.

The Control Terminal:

When a terminal file is opened, it causes the process to wait until a connection is established. In practice, user programs seldom open these files; they are opened by getty(8) or rlogind(8C) and become a user's standard input and output file.

If a process which has no control terminal opens a terminal file, then that terminal file becomes the control terminal for that process. The control terminal is thereafter inherited by a child process during a fork(2), even if the control terminal is closed.

The file |dev|tty is, in each process, a synonym for a control terminal associated with that process. It is useful for programs that wish to be sure of writing messages on the terminal no matter how output has been redirected. It can also be used for programs that demand a file name for output, when typed output is desired and it is tiresome to find out which terminal is currently in use.

A process can remove the association it has with its controlling terminal by opening the file /dev/tty and issuing an

```
ioctl(f, TIOCNOTTY, 0);
```

This is often desirable in server processes.

Process Groups:

Command processors such as csh(1) can arbitrate the terminal between different jobs by placing related jobs in a single process group and associating this process group with the terminal. A terminal's associated process group may be set using the TIOCSPGRP ioctl(2):

ioctl(fildes, TIOCSPGRP, &pgrp);

or examined using TIOCGPGRP, which returns the current process group in pgrp. The new terminal driver aids in this arbitration by restricting access to the terminal by processes which are not in the current process group; see "Job Access Control" below.

Modes:

The terminal drivers have three major modes, characterized by the amount of processing on the input and output characters:

The normal mode. In this mode lines of input are collected and input editing is done. The edited line is made available when it is completed by a newline, or when the t_brkc character (normally undefined) or t_eofc character (normally an EOT, control-D, hereafter ^D) is entered. A carriage return is usually made synonymous with newline in this mode, and replaced with a newline whenever it is typed. All driver functions (input editing, interrupt generation, output processing such as delay generation and tab expansion, etc.) are available in this mode.

CBREAK This mode eliminates the character, word, and line editing input facilities, making the input character available to the user program as it is typed. Flow control, literal-next and interrupt processing are still done in this mode. Output processing is done.

RAW This mode eliminates all input processing and makes all input characters available as they are typed; no output processing is done either.

The style of input processing can also be very different when the terminal is put in non-blocking I/O mode; see the FNDELAY flag described in *fcntl(2)*. In this case a *read(2)* from the control terminal will never block, but rather return an error indication (EWOULDBLOCK) if there is no input available.

A process may also request that a SIGIO signal be sent it whenever input is present and also whenever output queues fall below the low-water mark. To enable this mode the FASYNC flag should be set using fcntl(2).

Input Editing:

A terminal connected to a UNIX operating system ordinarily operates in full-duplex mode. Characters may be typed at any time, even while output is occurring, and are only lost when the system's character input buffers become completely choked, which is rare, or when the user has accumulated the maximum allowed number of input characters that have not yet been read by some program. Currently this limit is 256 characters. In RAW mode, the terminal driver throws away all input and output without notice when the limit is reached. In CBREAK or cooked mode it refuses to accept any further input and, if in the new line discipline, rings the terminal bell.

Input characters are normally accepted in either even or odd parity with the parity bit being stripped off before the character is given to the program. By clearing either the EVEN or ODD bit in the flags word it is possible to have input characters with that parity discarded (see "Summary," below.

In all of the line disciplines, it is possible to simulate terminal input using the TIOCSTI ioctl, which takes, as its third argument, the address of a character. The system pretends that this character was typed on the argument terminal, which must be the control terminal except for the super-user. (This call is not in standard Version 7 UNIX operating systems).

Input characters are normally echoed by putting them in an output queue as they arrive. This may be disabled by clearing the ECHO bit in the flags word using the stty(3C) call or the TIOCSETN or TIOCSETP ioctls (see "Summary," below).

In cooked mode, terminal input is processed in units of lines. A program attempting to read will normally be suspended until an entire line has been received (but see the description of SIGTTIN in "Job Access Control" and of FIONREAD in "Summary," both below.) No matter how many characters are requested in the read call, at most one line will be returned. It is not, however, necessary to read a whole line at once; any number of characters may be requested in a read -- even one -- without losing information.

During input, line editing is normally done, with the erase character sg_erase (by default, DELETE) logically erasing the last character typed and the sg_kill character (default, ^U: control-U) logically erasing the entire current input line. These characters never erase beyond the beginning of the current input line or an eof. These characters may be entered literally by preceding them with "\"; the "\" will normally be erased when the character is typed.

The drivers normally treat either a carriage return or a newline character as terminating an input line, replacing the return with a newline and echoing a return and a line feed. If the CRMOD bit is cleared in the local mode word then the processing for carriage return is disabled, and it is simply echoed as a return, and does not terminate cooked mode input.

In the new driver there is a literal-next character (normally ^V) which can be typed in both cooked and CBREAK mode preceding any character to prevent its special meaning to the terminal handler. This is to be preferred to the use of "\" escaping erase and kill characters, but "\" is retained with its old function in the new line discipline.

The new terminal driver also provides two other editing characters in normal mode. The worderase character, normally 'W, erases the preceding word, but not any spaces before it. For the purposes of 'W, a word is defined as a sequence of non-blank characters, with tabs counted as blanks. Finally, the reprint character, normally 'R, retypes the pending input beginning on a new line. Retyping occurs automatically in cooked mode if characters which would normally be erased from the screen are fouled by program output.

Input echoing and Redisplay:

The terminal driver has several modes (not present in standard UNIX Version 7 operating systems) for handling the echoing of terminal input, controlled by bits in a local mode word.

Hardcopy terminals. When a hardcopy terminal is in use, the LPRTERA bit is normally set in the local mode word. Characters which are logically erased are then printed out backwards preceded by "\" and followed by "\" in this mode.

CRT terminals. When a CRT terminal is in use, the LCRTBS bit is normally set in the local mode word. The terminal driver then echoes the proper number of erase characters when input is erased; in the normal case where the erase character is a ^H this causes the cursor of the terminal to back up to where it was before the logically erased character was typed. If the input has become fouled due to interspersed asynchronous output, the input is automatically retyped.

Erasing characters from a CRT. When a CRT terminal is in use, the LCRTERA bit may be set to cause input to be erased from the screen with a "backspace-space-backspace" sequence when character or word deleting sequences are used. A LCRTKILL bit may be set as well, causing the input to be erased in this manner on line kill sequences as well.

Echoing of control characters. If the LCTLECH bit is set in the local state word, then non-printing (control) characters are normally echoed as ^X (for some X) rather than being echoed unmodified; delete is echoed as ^?.

The normal modes for use on CRT terminals are speed dependent. At speeds less than 1200 baud, the LCRTERA and LCRTKILL processing is painfully slow, and stty(1) normally just sets LCRTBS and LCTLECH; at speeds of 1200 baud or greater all of these bits are normally set.

Stty(1) summarizes these option settings and the use of the new terminal driver as "newert."

Output Processing:

When one or more characters are written, they are actually transmitted to the terminal as soon as previously-written characters have finished typing. (As noted above, input characters are normally echoed by putting them in the output queue as they arrive.) When a process produces characters more rapidly than they can be typed, it will be suspended when its output queue exceeds some limit. When the queue has drained down to some threshold the program is resumed. Even parity is normally generated on output. The EOT character is not transmitted in cooked mode to prevent terminals that respond to it from hanging up; programs using RAW or CBREAK mode should be careful.

The terminal drivers provide necessary processing for cooked and CBREAK mode output including delay generation for certain special characters and parity generation. Delays are available after backspaces ^H, form feeds ^L, carriage returns ^M, tabs ^I and newlines ^J. The driver will also optionally expand tabs into spaces, where the tab stops are assumed to be set every eight columns, and optionally convert newlines to carriage returns followed by newline. These functions are controlled by bits in the tty flags word; see "Summary," below.

The terminal drivers provide for mapping between upper and lower case on terminals lacking lower case, and for other special processing on deficient terminals.

Finally, in the new terminal driver, there is a output flush character, normally ^O, which sets the LFLUSHO bit in the local mode word, causing subsequent output to be flushed until it is cleared by a program or more input is typed. This character has effect in both cooked and CBREAK modes and causes pending input to be retyped if there is any pending input. An *ioctl* to flush the characters in the input or output queues, TIOCFLUSH, is also available.

Upper-case Terminals and Hazeltines:

If the LCASE bit is set in the tty flags, then all upper-case letters are mapped into the corresponding lower-case letter. The upper-case letter may be generated by preceding it by "\". Upper case letters are preceded by a "\" when output. In addition, the following escape sequences can be generated on output and accepted on input:

To deal with Hazeltine terminals, which do not understand that ~ has been made into an ASCII character, the LTILDE bit may be set in the local mode word; in this case the character ~ will be replaced with the character ' on output.

Flow Control:

There are two characters (the stop character, normally ^S, and the start character, normally ^Q) which cause output to be suspended and resumed respectively. Extra stop characters typed when output is already stopped have no effect, unless the start and stop characters are made the same, in which case output resumes.

A bit in the flags word may be set to put the terminal into TANDEM mode. In this mode the system produces a stop character (default ^S) when the input queue is in danger of overflowing, and a start character (default ^Q) when the input has drained sufficiently. This mode is useful when the terminal is actually another machine that obeys those conventions.

Line Control and Breaks:

There are several *ioctl* calls available to control the state of the terminal line. The TIOCSBRK *ioctl* will set the break bit in the hardware interface causing a break condition to exist; this can be cleared (usually after a delay with *sleep*(3)) by TIOCCBRK. Break conditions in the input are reflected as a null character in RAW mode or as the interrupt character in cooked or CBREAK mode. The TIOCCDTR *ioctl* will clear the data terminal ready condition; it can be set again by

TIOCSDTR.

When the carrier signal from the dataset drops (usually because the user has hung up his terminal) a SIGHUP hangup signal is sent to the processes in the distinguished process group of the terminal; this usually causes them to terminate. The SIGHUP can be suppressed by setting the LNOHANG bit in the local state word of the driver. Access to the terminal by other processes is then normally revoked, so any further reads will fail, and programs that read a terminal and test for end-of-file on their input will terminate appropriately.

It is possible to ask that the phone line be hung up on the last close with the TIOCHPCL ioctl; this is normally done on the outgoing lines and dialups.

Interrupt Characters:

There are several characters that generate interrupts in cooked and CBREAK mode; all are sent to the processes in the control group of the terminal, as if a TIOCGPGRP *ioctl* were done to get the process group and then a *killpg*(2) system call were done, except that these characters also flush pending input and output when typed at a terminal (similar to TIOCFLUSH). The characters shown here are the defaults; the field names in the structures (given below) are also shown. The characters may be changed.

- ^C t_intrc (ETX) generates a SIGINT signal. This is the normal way to stop a process which is no longer interesting, or to regain control in an interactive program.
- ^\ t_quite (FS) generates a SIGQUIT signal. This is used to cause a program to terminate and produce a core image, if possible, in the file core in the current directory.
- ² t_suspc (EM) generates a SIGTSTP signal, which is used to suspend the current process group.
- 'Y t_dsuspc (SUB) generates a SIGTSTP signal as 'Z does, but the signal is sent when a program attempts to read the 'Y, rather than when it is typed.

Job Access Control:

When using the new terminal driver, if a process which is not in the distinguished process group of its control terminal attempts to read from that terminal its process group is sent a SIGTTIN signal. This signal normally causes the members of that process group to stop. If, however, the process is ignoring SIGTTIN, has SIGTTIN blocked, or is in the middle of process creation using vfork(2), the read will return -1 and set errno to EIO.

When using the new terminal driver with the LTOSTOP bit set in the local modes, a process is prohibited from writing on its control terminal if it is not in the distinguished process group for that terminal. Processes which are blocking or ignoring SIGTTOU signals or which are in the middle of a vfork(2) are excepted and allowed to produce output.

Terminal/Window Sizes:

In order to accommodate terminals and workstations with variable-sized windows, the terminal driver provides a mechanism for obtaining and setting the current terminal size. The driver does not use this information internally, but only stores it and provides a uniform access mechanism. When the size is changed, a SIGWINCH signal is sent to the terminal's process group so that knowledgeable programs may detect size changes. This facility was added in 4.3BSD and is not available in earlier versions of the system.

Summary of Modes:

Unfortunately, due to the evolution of the terminal driver, there are 4 different structures which contain various portions of the driver data. The first of these (sgttyb) contains that part of the information largely common between Version 6 and Version 7 UNIX operating systems. The second contains additional control characters added in Version 7. The third is a word of local state added in 4BSD, and the fourth is another structure of special characters added for the new

driver. In the future a single structure may be made available to programs which need to access all this information; most programs need not concern themselves with all this state.

Basic modes: sgtty.

The basic *ioctls* use the structure defined in $\langle sgtty.h \rangle$:

```
struct sgttyb {
    char sg_ispeed;
    char sg_ospeed;
    char sg_erase;
    char sg_kill;
    short sg_flags;
};
```

The sg_ispeed and sg_ospeed fields describe the input and output speeds of the device according to the following table, which corresponds to the DEC DH-11 interface. If other hardware is used, impossible speed changes are ignored. Symbolic values in the table are as defined in < sgtty.h>.

```
B0
            (hang up dataphone)
B50
       1
            50 baud
B75
       2
            75 baud
            110 baud
B110
       3
B134
       4
            134.5 baud
B150
       5
            150 baud
B200
       6
            200 baud
B300
       7
            300 baud
B600
            600 baud
B1200
       9
            1200 baud
B1800 10
            1800 baud
B2400 11
            2400 baud
B4800 12
            4800 baud
            9600 baud
B9600 13
EXTA 14
            19200 baud
EXTB 15
            (not supported)
```

Code conversion and line control required for IBM 2741s (134.5 baud) must be implemented by the user's program. The half-duplex line discipline required for the 202 dataset (1200 baud) is not supplied; full-duplex 212 datasets work fine.

The sg_erase and sg_kill fields of the argument structure specify the erase and kill characters respectively. (Defaults are DELETE and ^U.)

The sg_flags field of the argument structure contains several bits that determine the system's treatment of the terminal:

```
ALLDELAY 0177400 Delay algorithm selection
BSDELAY 0100000 Select backspace delays (not implemented):
BS<sub>0</sub>
             0
BS1
             0100000
VTDELAY
             0040000 Select form-feed and vertical-tab delays:
FF0
             0
FF1
             0040000
CRDELAY 0030000 Select carriage-return delays:
CR0
             0010000
CR1
CR2
             0020000
CR3
             0030000
TBDELAY 0006000 Select tab delays:
```

TAB0	0
TAB1	0021000
TAB2	0004000
XTABS	0006000
NLDELAY	0001400 Select new-line delays:
NL0	0
NL1	0000400
NL2	0001000
NL3	0001400
EVENP	0000200 Even parity allowed on input
ODDP	0000100 Odd parity allowed on input
RAW	0000040 Raw mode: wake up on all characters, 8-bit interface
CRMOD	0000020 Map CR into LF; output LF as CR-LF
ECHO	0000010 Echo (full duplex)
LCASE	0000004 Map upper case to lower on input and lower to upper on output
CBREAK	0000002 Return each character as soon as typed
TANDEM	0000001 Automatic flow control

The delay bits specify how long transmission stops to allow for mechanical or other movement when certain characters are sent to the terminal. In all cases a value of 0 indicates no delay.

Backspace delays are currently ignored but might be used for Terminet 300s.

If a form-feed/vertical tab delay is specified, it lasts for about two seconds.

Carriage-return delay type 1 lasts about .08 seconds and is suitable for the Terminet 300. Delay type 2 lasts about .16 seconds and is suitable for the VT05 and the TI 700. Delay type 3 is suitable for the concept-100 and pads lines to be at least 9 characters at 9600 baud.

New-line delay type 1 is dependent on the current column and is tuned for Teletype model 37's. Type 2 is useful for the VT05 and is about .10 seconds. Type 3 is unimplemented and is 0.

Tab delay type 1 is dependent on the amount of movement and is tuned to the Teletype model 37. Type 3, called XTABS, is not a delay at all but causes tabs to be replaced by the appropriate number of spaces on output.

The flags for even and odd parity control parity checking on input and generation on output in cooked and CBREAK mode (unless LPASS8 is enabled, see below). Even parity is generated on output unless ODDP is set and EVENP is clear, in which case odd parity is generated. Input characters with the wrong parity, as determined by EVENP and ODDP, are ignored in cooked and CBREAK mode.

RAW disables all processing save output flushing with LFLUSHO; full 8 bits of input are given as soon as it is available; all 8 bits are passed on output. A break condition in the input is reported as a null character. If the input queue overflows in raw mode all data in the input and output queues are discarded; this applies to both new and old drivers.

CRMOD causes input carriage returns to be turned into new-lines, and output and echoed new-lines to be output as a carriage return followed by a line feed.

CBREAK is a sort of half-cooked (rare?) mode. Programs can read each character as soon as typed, instead of waiting for a full line; all processing is done except the input editing: character and word erase and line kill, input reprint, and the special treatment of \ and EOT are disabled.

TANDEM mode causes the system to produce a stop character (default ^S) whenever the input queue is in danger of overflowing, and a start character (default ^Q) when the input queue has drained sufficiently. It is useful for flow control when the 'terminal' is really another computer which understands the conventions.

Note: The same "stop" and "start" characters are used for both directions of flow control; the t_stopc character is accepted on input as the character that stops output and is produced on output as the character to stop input, and the t_startc character is accepted on input as the character that restarts output and is produced on output as the character to restart input.

Basic ioctls

A large number of *ioctl*(2) calls apply to terminals. Some have the general form:

#include < sgtty.h >

ioctl(fildes, code, arg)

struct sgttyb *arg;

The applicable codes are:

TIOCGETP Fetch the basic parameters associated with the terminal, and store in the pointed-to sgttyb structure.

TIOCSETP Set the parameters according to the pointed-to sgttyb structure. The interface delays until output is quiescent, then throws away any unread characters, before changing the modes.

TIOCSETN Set the parameters like TIOCSETP but do not delay or flush input. Input is not preserved, however, when changing to or from RAW.

With the following codes arg is ignored.

TIOCEXCL Set "exclusive-use" mode: no further opens are permitted until the file has been closed.

TIOCNXCL Turn off "exclusive-use" mode.

TIOCHPCL When the file is closed for the last time, hang up the terminal. This is useful when the line is associated with an ACU used to place outgoing calls.

With the following codes arg is a pointer to an int.

TIOCGETD arg is a pointer to an int into which is placed the current line discipline number.

TIOCSETD arg is a pointer to an int whose value becomes the current line discipline number.

TIOCFLUSH If the int pointed to by arg has a zero value, all characters waiting in input or output queues are flushed. Otherwise, the value of the int is for the FREAD and FWRITE bits defined in < sys/file.h >; if the FREAD bit is set, all characters waiting in input queues are flushed, and if the FWRITE bit is set, all characters waiting in output queues are flushed.

The remaining calls are not available in vanilla Version 7 UNIX operating systems. In cases where arguments are required, they are described; arg should otherwise be given as 0.

TIOCSTI the argument points to a character which the system pretends had been typed on the terminal.

TIOCSBRK the break bit is set in the terminal.

TIOCCBRK the break bit is cleared.

TIOCSDTR data terminal ready is set.

TIOCCDTR data terminal ready is cleared.

TIOCSTOP output is stopped as if the "stop" character had been typed.

TIOCSTART output is restarted as if the "start" character had been typed.

TIOCGPGRP	arg is a pointer to an int into which is placed the process group ID of the process group for which this terminal is the control terminal.
TIOCSPGRP	arg is a pointer to an int which is the value to which the process group ID for this terminal will be set.
TIOCCONS	Forward messages written to /dev/console to this tty device.
TIOCOUTQ	returns in the int pointed to by arg the number of characters queued for output to the terminal.
FIONREAD	returns in the int pointed to by arg the number of characters immediately readable from the argument descriptor. This works for files, pipes, and terminals.

Tchars

The second structure associated with each terminal specifies characters that are special in both the old and new terminal interfaces: The following structure is defined in $\langle sys/ioctl.h \rangle$, which is automatically included in $\langle systy.h \rangle$:

```
struct tchars {
                                   /* interrupt */
        char
                 t intre;
                                   /* quit */
        char
                 t_quitc;
                 t startc:
                                   /* start output */
        char
                                   /* stop output */
                 t stope;
         char
                 t eofc;
                                   /* end-of-file */
        char
                                   /* input delimiter (like nl) */
         char
                 t brkc;
};
```

The default values for these characters are $^{\circ}$ C, $^{\circ}$, $^{\circ}$ Q, $^{\circ}$ S, $^{\circ}$ D, and 0xff. A character value of 0xff eliminates the effect of that character. The t_brkc character, by default 0xff, acts like a new-line in that it terminates a 'line,' is echoed, and is passed to the program. The 'stop' and 'start' characters may be the same, to produce a toggle effect. It is probably counterproductive to make other special characters (including erase and kill) identical. The applicable *ioctl* calls are:

TIOCGETC

Get the special characters and put them in the specified structure.

TIOCSETC Set the special characters to those given in the structure.

Local mode

The third structure associated with each terminal is a local mode word. The bits of the local mode word are:

```
LCRTBS
               000001 Backspace on erase rather than echoing erase
LPRTERA
               000002 Printing terminal erase mode
LCRTERA
               000004 Erase character echoes as backspace-space-backspace
               000010 Convert ~ to ' on output (for Hazeltine terminals)
LTILDE
LMDMBUF
               000020 Stop/start output when carrier drops
               000040 Suppress output translations
LLITOUT
               000100 Send SIGTTOU for background output
LTOSTOP
LFLUSHO
               000200 Output is being flushed
LNOHANG
               000400 Don't send hangup when carrier drops
LETXACK
               001000 Diablo style buffer hacking (unimplemented)
               002000 BS-space-BS erase entire line on line kill
LCRTKIL
LPASS8
               004000 Pass all 8 bits through on input, in any mode
LCTLECH
               010000 Echo input control chars as ^X, delete as ^?
LPENDIN
               020000 Retype pending input at next read or input character
LDECCTO
               040000 Only 'Q restarts output after 'S, like DEC systems
LNOFLSH
               100000 Inhibit flushing of pending I/O when an interrupt character is typed.
```

The applicable *ioctl* functions are:

TIOCLBIS arg is a pointer to an int whose value is a mask containing the bits to be set in the local mode word.

TIOCLBIC arg is a pointer to an int whose value is a mask containing the bits to be cleared

in the local mode word.

TIOCLSET arg is a pointer to an int whose value is stored in the local mode word.

TIOCLGET arg is a pointer to an int into which the current local mode word is placed.

Local special chars

The final control structure associated with each terminal is the *ltchars* structure which defines control characters for the new terminal driver. Its structure is:

```
struct Itchars {
        char
                                   /* stop process signal */
                 t_suspc;
                                   /* delayed stop process signal */
                 t dsuspc;
        char
        char
                 t rprntc;
                                   /* reprint line */
                                   /* flush output (toggles) */
        char
                 t flushc;
        char
                 t werasc;
                                   /* word erase */
        char
                 t_inextc;
                                   /* literal next character */
};
```

The default values for these characters are ^Z, ^Y, ^R, ^O, ^W, and ^V. A value of 0xff disables the character.

The applicable *ioctl* functions are:

TIOCSLTC arg is a pointer to an *ltchars* structure which defines the new local special characters.

TIOCGLTC

arg is a pointer to an *ltchars* structure into which is placed the current set of local special characters.

Window/terminal sizes

Each terminal has provision for storage of the current terminal or window size in a winsize structure, with format:

```
struct winsize {
    unsigned short ws_row; /* rows, in characters */
    unsigned short ws_col; /* columns, in characters */
    unsigned short ws_xpixel; /* horizontal size, pixels */
    unsigned short ws_ypixel; /* vertical size, pixels */
};
```

A value of 0 in any field is interpreted as "undefined;" the entire structure is zeroed on final close.

The applicable *ioctl* functions are:

TIOCGWINSZ

arg is a pointer to a struct winsize into which will be placed the current terminal or window size information.

TIOCSWINSZ

arg is a pointer to a struct winsize which will be used to set the current terminal or window size information. If the new information is different than the old information, a SIGWINCH signal will be sent to the terminal's process group.

```
FILES
```

/dev/tty /dev/tty* /dev/console

SEE ALSO

csh(1), stty(1), tset(1), ioctl(2), sigvec(2), stty(3C), getty(8)

UN(4)

NAME

un - IBM RT PC Baseband Adapter for use with Ethernet

SYNOPSIS

For the IBM RT PC:

device un0 at iocc0 csr 0xf4080000 priority 5 device un1 at iocc0 csr 0xf4088000 priority 4

For the IBM 6152 Academic System:

device un0 at iocc0 csr 0x00001550 priority 3 device un0 at iocc0 csr 0x000015511 priority 11

DESCRIPTION

The un interface provides access to a 10 Mb/s Ethernet network through an IBM RT PC Baseband Adapter for use with Ethernet.

The hardware has 16 kilobytes of dual-ported RAM. Four kilobytes are used for transmit buffers and 12k for receive buffers. The 4k are used to provide two 2k transmit buffers. The hardware permits the processor to fill one buffer while the board is transmitting from the other. The 12k-segment takes the form of 96 128-byte pages. Packets longer than 128 bytes can span multiple pages. The processor and the adapter use two registers to maintain a circular buffer of received packets. The driver writes one register to inform the adapter where the "full" pages start (so that the board doesn't refill them before the driver empties them). The driver reads the other register to determine where the empty pages start (so that it doesn't read empty pages before the board fills them). The hardware permits the processor to read from one part of the circular buffer while the board is receiving packets into another part of the buffer.

The host's Internet address is specified at boot time with an SIOCSIFADDR ioctl. The *un* interface employs the address resolution protocol described in arp(4P) to map dynamically between Internet and Ethernet addresses on the local network.

The interface supports the "trailer" protocol used by VAXen to minimize byte-shuffling in processing received packets. The use of trailers is negotiated with ARP. This negotiation may be disabled on a per-interface basis by setting the IFF_NOTRAILERS flag with an SIOCSIFFLAGS ioctl.

WARNING(fortheIBMRTPC The IBM Ethernet adapter requires dynamic RAM refresh. Do not place the card in slot 5 in the desk-model IBM 6151 Model 10, 15, 115 RT PC, or in slot 8 in the floor-model IBM 6150 Model 20, 25, 125 RT PC; those slots do not provide dynamic RAM refresh.

NOTE

This interface also supports the Ungermann-Bass Personal Computer Network Interface Controller, model number 2274A (PC-NIC card).

The adapter addresses 0xf4090000 and 0xf4098000 are also acceptable default addresses for compatibility with AIX (Advanced Interactive Executive) which uses those addresses. Users expecting to have AIX in co-residence should use those addresses.

DIAGNOSTICS

un%d: huge packet. The hardware detected an illegally large packet and truncated it.

un%d: ethernet jammed. The hardware encountered 16 consecutive collisions in attempting to transmit a packet using the Ethernet exponential backoff algorithm. The packet was dropped. (IBM RT PC only)

un%d: ethernet not responding (is it connected?). The hardware determined that the Ethernet was shorted or disconnected.

UN(4)

un%d: can't handle af%d. The interface was handed a message with addresses formatted in an unsuitable address family; the packet was dropped.

SEE ALSO

intro(4N), inet(4F), arp(4P)

VGA(4) VGA(4)

NAME

vga - IBM Video Graphics Array planar adapter device driver for the ibm8503, ibm8513, ibm8514, ibm8604

SYNOPSIS

pseudo-device vga

DESCRIPTION

The Video Graphics Array is an display adapter that resides on the planar of the PC/2 systems. It is capable of driving many models of IBM displays. All modes of the adapter are supported, but the type of display attached will limit which modes are available.

The programming interface is a mix of direct memory management and PC BIOS, thus enabling the graphic modes to be supported.

The following is the available modes for the vga device driver. The initial default mode is mode 03h (720x400 pels).

Mode	Туре	Colors/ Shades	Alpha Format	Buffer Start	Box Size	Max Pages	PELs
H 00	A/N	16/256K	40x25	B8000	8x8	8	320x200
* H00	A/N	16/256K	40x25	B8000	8x14	8	320x350
+ H00	A/N	16/256K	40x25	B8000	9x16	8	360x400
01 H	A/N	16/256K	40x25	B8000	8x8	8	320x200
01H *	A/N	16/256K	40x25	B8000	8x14	8	320x350
01H +	A/N	16/256K	40x25	B8000	9x16	8	360x400
02H	A/N	16/256K	80x25	B8000	8x8	8	640x200
02H *	A/N	16/256K	80x25	B8000	8x14	8	640x350
02H +	A/N	16/256K	80x25	B8000	9x16	8	720x400
03H	A/N	16/256K	80x25	B8000	8x8	8	720x400
03H *	A/N	16/256K	80x25	B8000	8x14	8	640x350
03H +	A/N	16/256K	80x25	B8000	9x16	8	720x400
04H	APA	4/256K	40x25	B8000	8x8	1	320x200
05H	APA	4/256K	40x25	B8000	8x8	1	320x200
06H	APA	2/256K	80x25	B8000	8x8	1	640x200
07H	A/N	4	80x25	B0000	9x14	8	720x350
07H +	A/N	4	80x25	B0000	9x16	8	720x400
0DH	APA	16/256K	40x25	A0000	8x8	8	320x200
0EH	APA	16/256K	80x25	A0000	8x8	4	640x200
0FH	APA	4	80x25	A0000	8x14	2	640x350
10H	APA	16/256K	80x25	A0000	8x14	2	640x350
11H	APA	2/256K	80x30	A0000	8x16	1	640x480
12H	APA	16/256K	80x30	A0000	8x16	1	640x480
13H	APA	256/256K	40x25	A0000	8x8	1	320x200
		•					

^{*} Extended Graphics Adapter text modes with 350 scan line

The default color table depends on the mode in use. Mode 3's default is:

0	black	8	gray
1	blue	9	light blue
2	green	10	light green
3	cyan	11	light cyan

^{+ 9}x16 enhanced text modes with 400 scan lines

4	red	12	light red
5	magenta	13	light magenta
6	brown	14	yellow
7	white	15	bright white

Modes can be switched using the < ESC > < command if you are using *ibmemul(4)*. When doing graphics, ioctls provided by *bufemul(4)* can be used to switch modes. See the *Personal System/2 Model 50 and 60 Technical Reference* for more information about the programming interface.

DIAGNOSTICS

None.

FILES

/dev/console /dev/ttyvga /dev/vga

SEE ALSO

cons(4), ibmemul(4), kbdemul(4), tty(4)
"IBM/4.3 Console Emulators" in Volume II, Supplementary Documents

BUGS

None.

XEMUL(4)

NAME

xemul - X input emulator for queuing keyboard and mouse events

SYNOPSIS

pseudo-device xemul

DESCRIPTION

Xemul is used for queuing input events from the keyboard and mouse into a queue area shared between the kernel and a user-level window manager (such as X).

Starting the X Emulator

Since xemul is a non-standard input emulator, the user should open the non-standard console device associated with the display being used, e.g. /dev/aed, /dev/apa8. After the open, the user should perform the EISETD ioctl command to set the E_XINPUT emulator. The following piece of code does this initialization:

```
#include < machinecons/screen_conf.h >
#include < machinecons/xio.h >
main()
{
    int fd, input_emul;

    fd = open ("/dev/apa16", O_RDWR);
    input_emul = E_XINPUT;
    ioctl (fd, EISETD, &input_emul);
}
```

Once you have set the input emulator to E_XINPUT, all input from the keyboard is queued in shared memory. The normal read system call returns an error. As described in the emulator document, when you open the non-standard device, the output emulator defaults to the buffer emulator. This assumes the user process takes over the display and any write system calls are buffered until the user process gives up the display; see bufemul(4). Also see stdemul(4) or ibmemul(4) if you wish to change the output emulator back to the standard emulator.

To forward mouse events to xemul, set the mouse device associated with the display to a line discipline which forwards to an input emulator. See tb(4) for more information. Because xemul is a non-standard emulator, the default emulators are restored automatically when the last process holding the display open dies or closes the device.

Data Structures

The following data structures are defined in < machinecons/xio.h > and in < machinecons/qevent.h > and describe the data shared between the kernel and the user-level process. (Since xio.h includes qevent.h, user programs need only include xio.h.)

```
typedef struct XIoAddr {
                                      /* Status of emulator (not used) */
       short status;
       XEvent *ibuff:
                                      /* Pointer to event queue */
                                      /* Circular queue size (power of 2) */
       int
              iqsize;
                                      /* Queue head */
       int
              ihead:
                                      /* Queue tail */
       int
              itail:
       XCursor mouse;
                                      /* Current Mouse position */
       XCursor hotspot;
                                      /* Current Mouse hot spot */
                                      /* Current Mouse movement box */
       XBox
                mbox;
                                      /* = 0 then make = 1 then break */
       int
             make break;
       short mthreshold;
                                      /* Mouse motion parameter */
```

```
short mscale;
                                      /* Mouse scale factor (if negative
                                         then do square).
                                      /* Hide mouse box */
       MSBox hmbox;
} XIoAddr;
typedef XIoAddr *XIoAddrAddr;
struct XBuffArea {
       XIoAddr xioa;
                                      /* Queue and control information */
       XEvent ibuff[XMAXEVQ];
                                      /* Circular event queue */
};
/* The event queue */
typedef struct X eventqueue {
       XEvent *events;
                                      /* input event buffer */
       int size;
                                      /* size of event buffer */
       int head;
                                      /* index into events */
                                      /* index into events */
       int tail;
} XEventQueue;
typedef struct X event {
       u_short xe_x;
                                      /* x position */
                                      /* y position */
       u short xe y;
                                      /* 10 millisecond units (button only) */
       u short xe time;
                                      /* button or motion? */
       u char xe type;
       u_char xe_key;
                                      /* the key (button only) */
       u char xe direction;
                                      /* which direction (button only) */
       u char xe device;
                                      /* which device (button only) */
} XEvent;
/* xe type field */
#define XE BUTTON
                               0
                                      /* button moved */
#define XE MMOTION
                                      /* mouse moved */
                               1
#define XE_TMOTION
                               2
                                      /* tablet moved */
/* xe direction field */
                                      /* up */
#define XE KBTUP
                               0
#define XE KBTDOWN
                               1
                                      /* down */
#define XE_KBTRAW
                               2
                                      /* undetermined */
/* xe device field */
#define XE MOUSE
                                      /* mouse */
                               1
#define XE DKB
                                      /* main keyboard */
#define XE_TABLET
                               3
                                      /* graphics tablet */
                                      /* auxiliary */
#define XE AUX
#define XE_CONSOLE
                                      /* console */
```

```
/* mouse motion rectangle */
typedef struct X box {
        short bottom;
        short right;
        short left;
        short top;
} XBox;
/* mouse cursor position */
typedef struct _X_cursor {
        short x;
        short y;
} XCursor;
/* Mouse locator bitmap */
typedef struct
        short data[16];
        short mask[16];
        struct {
                short v, h;
        } hotSpot;
} QIOLocator;
typedef struct {
        short bottom;
        short top;
        short left;
        short right;
                        /* 0 - not active, 1 - active */
        int
              flags;
} MSBox;
```

Shared Memory

Once the user process sets xemul, the following code should be performed to get the address of the shared memory:

```
#include < machinecons/xio.h >
XIoAddrAddr XAddr;
XEventQueue *queue;
ioctl (fd, QIOCADDR, &XAddr);
queue = (XEventQueue *) (&XAddr->ibuff);
```

The last assignment above creates a pointer to the section of the shared memory where the queue pointers and information are kept.

Event Queue

At the start, the *head* and *tail* indices in the *XEventQueue* are both zero. When an event occurs, the information is stored at the *tail*, and the index is bumped up by one. If the *tail* index has reached the end of the queue, it wraps around to zero (circular queue). All input events are ignored if the *tail* index catches up to the *head* index.

XEMUL(4) XEMUL(4)

The user should poll or issue a select to find out when one event occurs (head != tail). The user should then process the information from the event pointed to by the *head* index and then adjust the *head* index in the same manor as that described for the *tail*. The following code is an example of this process:

Event Data

The user need only to analyze the data in the event passed to determine what has occurred. The xe_x and xe_y always contain the current mouse (x, y) position for any event. This same information is also always available in the mouse entry in the XIoAddr structure. The xe_t entry is a timestamp (in 10 millisecond units) marking that event. The xe_t entry describes what type of event occurred, XE_BUTTON meaning a keyboard key or mouse button event, XE_MMOTION meaning a mouse motion event, and XE_TMOTION meaning a tablet motion event. Only if the event was an XE_BUTTON will the xe_t key, xe_t direction, and xe_t device fields be filled in accordingly. The xe_t device tells you whether the event was from the keyboard, mouse, tablet, etc. The xe_t contains the keyboard key code or describes the mouse button. The xe_t direction tells you whether the key/button was depressed or released (DOWN or UP). Button events are always presented as new events. See hardware documentation for the keyboard codes and tb(4) for the mouse-button report.

Motion Events

Motion events are joined if the previous event was a motion and the user had not yet read it. As stated before, the current mouse position is always kept in the shared memory structure XloAddr.mouse. Motion events are not always reported as events to the user. The user may set the mouse motion box, XloAddr.mbox, to a rectangle in which motion events should not be reported. This is a key feature of xemul, which optimizes events to those about which the user cares. This is possible because xemul tracks the mouse with the cursor/locator on the screen being used. Not being responsible for tracking the mouse on the screen, the user doesn't need to know every motion event.

Cursor/Locator Control

The user can perform the following ioctls for controlling the mouse cursor tracking.

QIOCADDR The user passes the address of an XloAddrAddr which is filled with the address in shared memory where the structure is found.

QIOCSMSTATE Set the mouse state. The user passes the XCursor structure to specify the

new mouse position.

XEMUL(4) XEMUL(4)

QIOCLDCUR The user passes a QIOLocator structure indicating what the cursor/locator

bitmap should be on the display.

QIOCHIDECUR This ioctl inhibits the cursor/locator from being displayed on the screen.

The cursor is still tracked and reported to the user.

QIOCSHOWCUR The user issues this ioctl to show the cursor/locator after it was hidden.

The locator appears at its current location (not where it was hidden). The

locator defaults to show.

QIOCSETSIZE Change the limits of the box in which the mouse tracks. Issuing this ioctl

turns off kernel mouse tracking and disables QIOCSHOWCUR.

QUICGETSIZE Return the current limits of the mouse box.

The user can also control the mouse cursor/locator by writing into the XloAddr shared memory structure. The following is a synopsis of each remaining part of this structure:

mouse

The current mouse position. XCursor structure.

hotspot

Current Mouse hot spot offset from the mouse position. XCursor structure.

mbox

Current Mouse movement box. XBox structure.

mthreshold

Mouse motion parameter (not used currently).

mscale

Mouse scale factor, if negative then do square (not used currently).

hmbox

Hide mouse box, which the user specifies to give xemul a rectangle in which it should hide the mouse. The emulator passes this box to the hardware locator routines mainly so software driven cursors do not display within this box. This is only used by displays with no hardware cursor support which need to know

when to get the software cursor out of the way.

Keyboard Ioctl Control

QIOCBELL Ring keyboard bell with integer volume passed from 0 (off) to 7 (loud).

QIOCCLICK

Set autokeyclick to integer volume passed between -1 (default), 0 (off) and

8 (loud).

QIOCAUTOREP

Turn keyboard keys autorepeat (1) on or (0) off, integer argument.

QIOCSETCAPSL

Turn on Caps Lock light on keyboard.

QIOCCLRCAPSL

Turn off Caps Lock light on keyboard.

QIOCSETNUML

Turn on Num Lock light on keyboard.

QIOCCLRNUML

Turn off Num Lock light on keyboard.

QIOCSETSCROLLL

Turn on Scroll Lock light on keyboard.

QIOCCLRSCROLLL

Turn off Scroll Lock light on keyboard.

FILES

/dev/aed IBM Academic Information Systems experimental display

/dev/apa8 IBM 6153 Monochrome Display

/dev/apa8c IBM 6154 Advanced Color Graphics Display

/dev/apa16 IBM 6155 Extended Monochrome Graphics Display

/dev/ega no kernel cursor tracking no kernel cursor tracking

XEMUL(4)

SEE ALSO

bufemul(4), ibmemul(4), kbdemul(4), mouse(4), stdemul(4), tb(4) "IBM/4.3 Console Emulators" in Volume II, Supplementary Documents

BUGS

Xemul does not implement threshold and scale features for the mouse.

Section 5. File Formats and Conventions

This section includes file formats and conventions. Man pages found in IBM/4.3, but not in 4.3BSD, are marked with an asterisk (*).

- a.out
- consoles*
- core
- dbx
- font3812*
- keyboard_codes*
- map3270
- printer3812*
- rc.config
- rvddb*
- rvdtab*

IBM/4.3 Manual Pages

A.OUT(5)

```
NAME
```

a.out - assembler and link editor output

SYNOPSIS

#include < a.out.h >

DESCRIPTION

A.out is the output file of the assembler as(1) and the link editor ld(1). Both programs make a.out executable if there were no errors and no unresolved external references. Layout information as given in the include file for the IBM RT PC is:

```
* Header prepended to each a out file.
*/
struct exec {
        long
               a_magic;
                                       /* magic number */
                                       /* size of text segment */
        unsigned
                       a text;
                                       /* size of initialized data */
        unsigned
                       a data;
                                       /* size of uninitialized data */
        unsigned
                       a bss;
                       a_syms;
        unsigned
                                       /* size of symbol table */
                                       /* entry point */
        unsigned
                       a entry;
                                       /* size of text relocation */
        unsigned
                       a trsize;
        unsigned
                       a drsize;
                                       /* size of data relocation */
};
#define OMAGIC
                       0407
                                       /* old impure format */
                                       /* read-only text */
#define NMAGIC
                       0410
#define ZMAGIC
                       0413
                                       /* demand load format */
* Macros which take exec structures as arguments and tell whether
* the file has a reasonable magic number or offsets to text | symbols | strings.
#define N BADMAG(x) \
   (((x).a magic)!=OMAGIC && ((x).a magic)!=NMAGIC && ((x).a magic)!=ZMAGIC)
#define N TXTOFF(x) \
        ((x).a \text{ magic} = ZMAGIC ? 2048 : size of (struct exec))
#define N SYMOFF(x) \
       (N TXTOFF(x) + (x).a text + (x).a data + (x).a trsize + (x).a drsize)
#define N STROFF(x) \
        (N SYMOFF(x) + (x).a syms)
```

The file has five sections: a header, the program text and data, relocation information, a symbol table and a string table (in that order). The last three may be omitted if the program was loaded with the -s option of ld(1) or if the symbols and relocation have been removed by strip(1).

In the header the sizes of each section are given in bytes. The size of the header is not included in any of the other sizes.

When an a.out file is executed, three logical segments are set up: the text segment, the data segment (with uninitialized data, which starts off as all 0, following initialized), and a stack. The text segment begins at 0 in the core image; the header is not loaded. If the magic number in the header is OMAGIC (0407), it means that the text segment is not to be write-protected and shared, so the data segment is immediately contiguous with the text segment. This is the oldest kind of executable program and is rarely used. If the magic number is NMAGIC (0410) or ZMAGIC (0413), the data segment is loaded at 0x10000000, and the text segment is not writable

A.OUT(5)

by the program; if other processes are executing the same file, they will share the text segment. For ZMAGIC format, the text segment begins at a 0 mod 2048-byte boundary in the a.out file, the remaining bytes after the header in the first block are reserved and should be zero. Here, the text and data sizes must both be multiples of 2048 bytes, and the pages of the file will be brought into the running image as needed, and not pre-loaded as with the other formats. This is especially suitable for large programs and is the default format produced by ld(1).

The user stack is located near the top of segment 1, after UPAGES of kernel stack and one page of redzone. The first page of user stack is reserved for kernel floating point use; therefore, the actual user stack starts at 0x1fffd800 (given that UPAGES = 3). The stack is automatically extended as required. The data segment is extended only as requested by brk(2).

After the header in the file follow the text, data, text relocation data relocation, symbol table and string table in that order. The text begins at the byte 2048 in the file for ZMAGIC format or just after the header for the other formats. The N_TXTOFF macro returns this absolute file position when given the name of an exec structure as argument. The data segment is contiguous with the text and immediately followed by the text relocation and then the data relocation information. The symbol table follows all this; its position is computed by the N_SYMOFF macro. Finally, the string table immediately follows the symbol table at a position that can be gotten easily using N_STROFF. The first 4 bytes of the string table are not used for string storage; instead they contain the size of the string table. The size includes the 4 bytes, and the minimum string table size is thus 4.

The layout of a symbol table entry and the principal flag values that distinguish symbol types are given in the include file as follows:

```
* Format of a symbol table entry.
*/
struct nlist {
        union {
            char
                      *n name; /* for use when in-core */
                                /* index into file string table */
            long
                      n strx;
        } n un;
                                /* type flag, i.e. N_TEXT etc; see below */
        unsigned char n type;
        char
                      n other;
                                /* see < stab.h > */
                      n desc;
        short
                                /* value of this symbol (or offset) */
        unsigned
                      n value;
#define n hash
                      n desc
                                /* used internally by ld */
* Simple values for n type.
#define N UNDF
                                /* undefined */
                      0x0
#define N ABS
                                /* absolute */
                      0x2
#define N TEXT
                      0x4
                                /* text */
#define N DATA
                      0x6
                                /* data */
#define N BSS
                      0x8
                                /* bss */
#define N COMM
                      0x12
                                /* common (internal to ld) */
#define N FN
                                 /* file name symbol */
                      0x1f
#define N EXT
                      01
                                /* external bit, or'ed in */
#define N TYPE
                                /* mask for all the type bits */
                      0x1e
/*
```

A.OUT(5) A.OUT(5)

```
* Other permanent symbol table entries have some of the N_STAB bits set.

* These are given in < stab.h >

*/
#define N_STAB 0xe0 /* if any of these bits set, don't discard */

/*

* Format for namelist values.

*/
#define N FORMAT "%08x"
```

In the a.out file a symbol's n_un.n_strx field gives an index into the string table. A n_strx value of 0 means that no name is associated with a particular symbol table entry. The field n_un.n_name can be used to refer to the symbol name only if the program sets this up using n strx and appropriate data from the string table.

If a symbol's type is undefined externally and the value field is non-zero, the loader *ld* interprets the symbol as the name of a common region whose size is shown by the value of the symbol.

The value of a byte in the text or data that is not a part of a reference to an undefined external symbol will appear in memory when the file is executed. If a byte in the text or data involves a reference to an undefined external symbol, as shown by the relocation information, then the value stored in the file is an offset from the associated external symbol. When the file is processed by the link editor and the external symbol becomes defined, the value of the symbol will be added to the bytes in the file.

If relocation information is present, it amounts to eight bytes per relocatable datum as in the following structure:

A datum with (r_pcrel && r_length = = 2) indicates branch address relocation; r_address points to the branch opcode. If the opcode is any of bb, bbx, bnb, bnbx, bali or balix, relocation is for a 20-bit pc-relative displacement field. If the opcode is bala or balax, relocation is for a 24-bit absolute displacement field.

A datum with $(r_{length} = 3)$ indicates split-address relocation on an instruction pair: either cal r,low; op r,high or cau r,high; op r,low. r_address points to the displacement field of the first instruction, the opcode of which indicates a high/low or low/high split address.

There is no relocation information if a_trsize + a_drsize = 0. If r_extern is 0, then r_symbolnum is really a n_type for the relocation. (e.g. N_TEXT meaning relative to segment text origin.)

SEE ALSO

```
adb(1), as(1), dbx(1), Id(1), nm(1), strip(1), stab(5)
```

BUGS

The size of the string table should be part of the header; many programs, however, assume the header is 32 bytes long. Such programs would have to be changed.

CONSOLES(5) CONSOLES(5)

NAME

consoles - utility database of display screens

DESCRIPTION

Consoles is a database containing one line for each display screen on the system. The first character of each line is an access code for use by the user. Following a blank, the name of the display screen device is listed. The allowable access codes and their meanings are as follows:

- 0 device not available to either the kernel or the user
- device available to the kernel as a console, but may not be opened by the user
- device not available to the kernel, but may be opened by the user
- device available both to the kernel and to the user
- x (any other character causes the line to be ignored as a comment)

A typical pair of entries might be

1 mono

2 aed

which would allow the user to keep console output on the IBM 5151 monochrome display, and reserve the IBM Academic Information Systems experimental display for application use.

Valid devices, and their product affiliation, are as follows:

mono IBM 5151 Monochrome Display

ega IBM 5154 Enhanced Graphics Display interface

apa8 IBM 6153 Advanced Monochrome Graphics Display

apa8c IBM 6154 Advanced Color Graphics Display

apa16 IBM 6155 Extended Monochrome Graphics Display

aed IBM Academic Information Systems experimental display

vga IBM PS/2 Display interface

ibm8514

IBM 8514/A Display interface

mpel IBM 5081 Display interface

Consoles is read by setscreen(8) when invoked by the user, such as from a .login file, or during boot (from /etc/rc.local).

FILES

/etc/consoles

SEE ALSO

ibm5081(4), ibm5151(4), ibm6153(4), ibm6154(4), ibm6155(4), ibm8514(4), ibm8514(4), vga(4), setscreen(8)

CORE(5)

NAME

core - format of memory image file

SYNOPSIS

#include < machine/param.h >

DESCRIPTION

The UNIX operating system writes out a memory image of a terminated process when any of various errors occur. See *sigvec*(2) for the list of reasons; the most common are memory violations, illegal instructions, bus errors and user-generated quit signals. The memory image is called "core" and is written in the process's working directory (provided it can be; normal access controls apply).

The maximum size of a core file is limited by setrlimit(2). Files which would be larger than the limit are not created.

The core file consists of the u area, whose size (in pages) is defined by the UPAGES manifest in the < machine/param.h> file. The u area starts with the process kernel stack and ends with the user structure as given in < sys/user.h>. The remainder of the core file consists first of the data pages and then the stack pages of the process image. The amount of data space image in the core file is given (in pages) by the variable u_dsize in the u area. The amount of stack image in the core file is given (in pages) by the variable u_ssize in the u area.

In general, the debugger adb(1) is sufficient to deal with core images.

SEE ALSO

adb(1), dbx(1), setrlimit(2), sigvec(2)

NAME

dbx - dbx symbol table information

DESCRIPTION

The compiler symbol information generated for dbx(1) uses the same structure as described in stab(5), with additional type and scope information appended to a symbol's name. The assembler directive used to describe symbol information has the following format:

.stabs "string", kind, 0, size, value

String contains the name, source language type, and scope of the symbol, kind specifies the memory class (e.g., external, static, parameter, local, register), and size specifies the byte size of the object, if relevant. The third field (0 above) is unused. For a global variable or a type, value is unused; for a local variable or parameter, it is the offset (defined below); for a register variable, it is the associated register number.

The different kinds of stab entries are interpreted by dbx as follows:

- N_GSYM The symbol is a global variable (e.g., .comm variable). The variable's address can be found from the corresponding ld(1) symbol entry, thus value for N_GSYM symbols is ignored. For example, a global variable "x" will have both an N_GSYM entry and an ld(1) entry (e.g., N_BSS + N_EXT). See a.out(5) for details about these other entries.
- N_FUN The symbol is a procedure or function. Size contains the line number of the entry point. Value contains the address of the entry point (in the text segment).

N STSYM

The symbol is a statically allocated variable for which an initial value has been specified. Value contains the address of the variable (in the data segment).

N LCSYM

The symbol is statically allocated, but not initialized.

- N RSYM The symbol is a register variable whose value is kept in the register denoted by value.
- N_PSYM The symbol is a parameter whose value may be pushed on the stack before the call, or may be passed in r2-r5. *Value* contains the offset from the base of the argument list (16 bytes below the top of the stack frame).
- N_LSYM The symbol is a local variable whose value is located in the most recently defined procedure's stack frame. *Value* is the offset (always negative) from the top of the local data area.

N PC, N MOD2

(Unsupported.) The symbol defines separate compilation information for pre-linking checking for Berkeley Pascal and Modula-2 programs respectively. For Pascal, the value field contains the line number that the symbol is defined on. The value field is not used for Modula-2.

N SO The symbol is a source file name. Size is 0 if compiled by pcc, 1 if by hc or pp.

The compilers order LBRAC, RBRAC, and symbol stabs by the following rules:

- Except for the following, every name scope is represented by an LBRAC/RBRAC pair:
 - File scope has no LBRAC/RBRAC pair.
 - Pcc omits the pair corresponding to a function definition (a function declaration followed by a function body).
 - Pcc omits the pair corresponding to the braces surrounding a function body, unless at least one declaration immediately follows the left brace.

Pcc places symbol stabs before the scope-opening LBRAC stab. Hc and pp place symbol stabs after the scope-opening LBRAC stab, matching the order of declarations and { }'s in the source.

Most of the source level information about a symbol is stored in the *string* field of the stab entry. Since strings are kept in a separate string table in the a.out file, they can be arbitrarily long. Thus there are no restrictions on the kind or length of information in the string field, and it was not necessary to modify the assembler or loader when extending or modifying the format of this information.

Below is a grammar describing the syntax of the symbol string. Except in the case of a constant whose value is a string, there are no blanks in a symbol string.

```
NAME:
               [a-zA-Z][a-zA-Z]
INTEGER:
               [-][0-9][0-9]*
REAL:
               [+-][0-9]*(.[0-9][0-9]*]([eE]([+-]])[0-9][0-9]*]
                                               -- \" represents "
STRING:
String:
  NAME ':' Class
  ':' Class
Class:
  'c' '= ' Constant ';'
  Variable
                               -- d, r, G, S, V, TypeId
                               -- f, F, J, P, Q
  Procedure
  Parameter
                               -- a, p, v, D, R
  NamedType
                               -- t, T
Constant:
   'i' INTEGER
   'r' REAL
  'c' OrdValue
   'b' OrdValue
   's' STRING
   'e' TypeId ',' OrdValue
   'S' TypeId ',' NumElements ',' NumBits ',' [01]*
OrdValue:
  INTEGER
NumElements:
  INTEGER
NumBits:
   INTEGER
Variable:
                -- local variable of type TypeId
   TypeId
   'd' Typeld
               -- floating register variable of type TypeId
   'r' TypeId
               -- register variable of type TypeId
   'S' TypeId
               -- module variable of type TypeId (static global in C)
   'V' TypeId -- own variable of type TypeId (static local in C)
   'G' TypeId -- global variable of type TypeId
```

```
Procedure:
   Proc
                               -- top level procedure
   Proc ',' NAME ',' NAME -- local to first NAME,
                               -- second NAME is corresponding ld symbol
Proc:
   'P'
               -- global procedure
   O,
               -- local procedure
  'F' TypeId -- function returning type TypeId
   'f' TypeId
               -- local function (static in C)
  'J' TypeId -- internal function
Parameter:
   'a' TypeId
              -- parameter passed by reference, in general register (Pascal only)
   'p' TypeId -- value parameter
   'v' TypeId -- parameter passed by reference (Pascal only)
  'D' TypeId -- value parameter in floating point register
   'R' TypeId -- value parameter in general register
NamedType:
   't' TypeId
               -- type name for type TypeId
   'T' TypeId -- C structure tag name for struct TypeId
TypeId:
  INTEGER
                               -- Unique (per compilation) number of type
                               -- Definition of type number
   INTEGER '=' TypeDef
  INTEGER '=' TypeAttrs TypeDef
-- Type attributes are extra information associated with a type,
-- such as alignment constraints or pointer checking semantics.
-- Dbx interprets some of these, but will ignore rather than complain
-- about any it does not recognize. Therefore this is a way to add
-- extra information for pre-linking checking.
TypeAttrs:
   '@' TypeAttrList ';'
TypeAttrList:
  TypeAttrList ',' TypeAttr
  TypeAttr
TypeAttr:
   'a' INTEGER
                               -- align boundary
  's' INTEGER
                               -- size in bits
  'p' INTEGER
                               -- pointer class (e.g., checking)
  BSTRING
                               -- something else
TypeDef:
  INTEGER
  Subrange
  Array
  Record
```

```
'e' EnumList ';'
                              -- enumeration
   '*' TypeId
                              -- pointer to TypeId
   'S' TypeId
                              -- set of TypeId
  'd' TypeId
                              -- file of TypeId
  ProcedureType
  'i' NAME ':' NAME ';'
                              -- imported type ModuleName:Name
  'o' NAME ';'
                                              -- opaque type
  'i' NAME ':' NAME ',' TypeId ';'
  'o' NAME ',' TypeId ';'
Subrange:
  'r' TypeId ';' INTEGER ';' INTEGER
   'a' TypeDef ';' TypeId
                              -- array [TypeD] of TypeId
  'A' TypeId
                              -- open array of TypeId
  'D' INTEGER ',' TypeId
                              -- N-dim. dynamic array
  'E' INTEGER ',' TypeId
                              -- N-dim. subarray
ProcedureType:
  'f' TypeId ';'
                              -- C function type
  'f' TypeId ',' NumParams ';' TParamList ';'
  'p' NumParams ';' TParamList ';'
NumParams:
  INTEGER
Record:
  's' ByteSize FieldList ';'
                              -- structure/record
  'u' ByteSize FieldList ';'
                              -- C union
ByteSize:
  INTEGER
FieldList:
  Field
  FieldList Field
Field:
  NAME ':' TypeId ',' BitOffset ',' BitSize ';'
BitSize:
  INTEGER
BitOffset:
  INTEGER
EnumList:
  Enum
  EnumList Enum
Enum:
  NAME ':' OrdValue ','
```

```
ParamList:
   Param
   ParamList Param
Param:
  NAME ':' TypeId ',' PassBy ';'
PassBy:
  INTEGER
TParam:
   TypeId ',' PassBy ';'
TParamList:
  TParam
   TParamList TParam
Export:
  INTEGER ExportInfo
ExportInfo:
   't' TypeId
  'f' TypeId ',' NumParams ';' ParamList ';'
   'p' NumParams ';' ParamList ';'
  'v' TypeId
   'c' '=' Constant
```

A '?' indicates that the symbol information is continued in the next stab entry. This directive can only occur where a ';' would otherwise separate the fields of a record or constants in an enumeration. It is useful when the number of elements in one of these lists is large.

FILES

stab.h

SEE ALSO

dbx(1), stab(5), a.out(5)

FONT3812(5) FONT3812(5)

NAME

font3812 - font structures for 3812 fonts

SYNOPSIS

#include < pmp/font3812.h >

DESCRIPTION

Font38/2.h defines the font structures that are built by cvt38/2(8) and width38/2(8). Three file formats are described. The .dat file contains the raster data for the font. The .ndx file contains an index by IBM character name. The codepage index file contains an index by troff(1) character name.

```
Format for the .dat font files for the 3812.
     This file contains the raster data for all the characters in the font.
     The .dat file contains the fixed length information (char info)
     for each character followed by the variable length raster
     data for that character. The height, width, a space,
     c space and offset are expressed in 240 points per inch.
     For fixed portion of the font information:
typedef struct {
   short x;
                         /* width of bit pattern*/
   short v:
                         /* height of bit pattern*/
                         /* space before bits*/
   short a space;
                         /* space following bits*/
   short c space;
   short offset;
                         /* baseline offset*/
} char_info;
     Each character contains char info plus raster data.
     The length of raster data is determined by:
          (x + 7) / 8 * y
 */
typedef struct {
   char info c data;
   char *r data;
                     /*pointer to raster bit pattern*/
} ibm fdata;
     Format for the .ndx file:
     This file is an index into the .dat file by IBM character name.
     The .ndx file contains one index data structure for each IBM
     character in the font.
     Each 8 byte name corresponds to an IBM character name.
     The offset indicates the byte offset into the font data for that character.
      The length indicates the length of the font data.
 */
typedef struct {
     char name[8];
                         /* IBM character name*/
     long offset;
                         /* offset into .dat file*/
     long length;
                         /* length of char info & raster data for this character */
} index data;
```

FONT3812(5)

```
Format for the codepage index into font data:
              A codepage identifies the set of characters to be used from a font.
              An IBM font contains characters with character names. For example:
              the letter "a" is LA010000 and the letter beta is GB010000.
              The codepage entry would select these two characters and give
              their ASCII name. Since beta is not directly represented in ASCII
              it is given its troff coded name (*b).
              The codepage would contain:
                   a LA010000
                   *b GB010000
              An offset file is built by width3812 for each font for a size
              and a particular codepage table. It is named ff.s.cpage, where
                         = the font family name
                           = the size
                    cpage
                                 = the name of the codepage
              For example, the offset file for the Sonoran Sans Serif (named ss) font
              using the stdcp codepage and size 6 is named: ss.6.stdcp
              The file has the following format:
                total chars: Integer count of characters in font.
                cp index
                                 : One entry for each character listed in the codepage.
        typedef struct {
                                 /* Characters are listed in same order*/
                                 /* as in the codepage table.*/
                           /* offset into the .dat file for char*/
           int offset;
                            /* length of char info and raster data for this character*/
           int length;
        } cp_index;
SEE ALSO
        cvt3812(8), width3812(8)
```

2

NAME

keyboard codes - keyboard scancode table

DESCRIPTION

The keyboard_codes file is read by the pf program and specifies the name associated with each scancode. It is also used to generate the scancode tables used by the kernel keyboard driver. There are several types of lines, identified by the first word of the line. Lines beginning with * are ignored as comments. Lines beginning with # are ignored by pf, but not by the kernel table building script (which produces |sys|machinecons|kbde codes.h).

function fn name

fn_name is the name of a meta-function.

codes values

specifies the number of modes to be specified for each scancode. This is 6 in the current implementation.

codelength length

specifies how many characters are stored in the primary table. It must be at least 2 (and is 2 in the current implementation).

type number "name"

specifies a name for the numeric type number. More than one name can be specified for a given number.

scancode meta fn_name

specifies that the hex scancode is bound directly to the keyboard meta-function fn name.

scancode normal shift capslock control alt action [name]

specifies the values in all six modes for the given hex scancode. The name associated with the scancode is the normal value unless the optional name is present.

FILES

/usr/lib/keyboard codes

SEE ALSO

pf(1), kbdemul(4)

MAP3270(5)

NAME

map3270 - data base for mapping ASCII keystrokes into IBM 3270 keys

SYNOPSIS

/etc/map3270

DESCRIPTION

When emulating IBM-style 3270 terminals under IBM/4.3 (see tn3270(1)), a mapping must be performed between sequences of keys hit on a user's (ascii) keyboard, and the keys that are available on a 3270. For example, a 3270 has a key labeled EEOF which erases the contents of the current field from the location of the cursor to the end. In order to accomplish this function, the terminal user and a program emulating a 3270 must agree on what keys will be typed to invoke the EEOF function.

The requirements for these sequences are:

- (1) That the first character of the sequence be outside of the standard ascii printable characters:
- (2) That no sequence be an initial part of another (although sequences may share initial parts).

FORMAT

The file consists of entries for various keyboards. The first part of an entry lists the names of the keyboards which use that entry. These names will often be the same as in /etc/termcap (see termcap(5)); however, note that often the terminals from various termcap entries will all use the same map3270 entry; for example, both 925 and 925vb (for 925 with visual bells) would probably use the same map3270 entry. Additionally, there are occasions when the terminal type defines a window manager, and it will then be necessary to specify a keyboard name (via the KEYBD environment variable) as the name of the entry. After the names, separated by vertical bars ('|'), comes a left brace ('{'}); the definitions; and, finally, a right brace ('{'}).

Each definition consists of a reserved keyword (see list below) which identifies the 3270 function (extended as defined below), followed by an equal sign ('='), followed by the various ways to generate this particular function, followed by a semi-colon (';'). Each way is a sequence of strings of *printable* ascii characters enclosed inside single quotes (""); various ways (alternatives) are separated by vertical bars ('|').

Inside the single quotes, a few characters are special. A caret ('^') specifies that the next character is the "control" character of whatever the character is. So, 'a' represents control-a, ie: hexadecimal 1 (note that 'A' would generate the same code). To generate rubout (DEL), one enters 'a'. To represent a control character inside a file requires using the caret to represent a control sequence; simply typing control-A will not work. Note: the ctrl-caret sequence (to generate a hexadecimal 1E) is represented as 'a' (not 'a').

In addition to the caret, a letter may be preceded by a backslash ('\'). Since this has little effect for most characters, its use is usually not recommended. For the case of a single quote ("'), the backslash prevents that single quote from terminating the string. For the case of a caret ('^'), the backslash prevents the caret from having its special meaning. To have the backslash be part of the string, it is necessary to place two backslashes ('\\') in the file.

In addition, the following characters are special:

'\E' means an escape character;

'\n' means newline:

'\t' means tab;

'\r' means carriage return.

It is not necessary for each character in a string to be enclosed within single quotes. '\E\E\E'

MAP3270(5) MAP3270(5)

means three escape characters.

Comments, which may appear anywhere on a line, begin with a hash mark ('#'), and terminate at the end of that line. However, comments cannot begin inside a quoted string; a hash mark inside a quoted string has no special meaning.

3270 KEYS SUPPORTED

The following is the list of 3270 key names that are supported in this file. Note that some of the keys don't really exist on a 3270. In particular, the developers of this file have relied extensively on the work at the Yale University Computer Center with their 3270 emulator which runs in an IBM Series/1 front end. The following list corresponds closely to the functions that the developers of the Yale code offer in their product.

In the following list, the starred (*) functions are not supported by tn3270(1). An unsupported function will cause tn3270(1) to send a (possibly visual) bell sequence to the user's terminal.

3270 Key Name **Functional Description** (*)LPRT local print DP dup character FM field mark character CURSEL cursor select **CENTSIGN** EBCDIC cent sign RESHOW redisplay the screen erase input **EINP EEOF** erase end of field DELETE delete character **INSRT** toggle insert mode **TAB** field tab field back tab **BTAB** COLTAB column tab **COLBAK** column back tab indent one tab stop INDENT undent one tab stop **UNDENT** new line NLHOME home the cursor UP up cursor **DOWN** down cursor **RIGHT** right cursor LEFT left cursor set a column tab **SETTAB** delete a columntab DELTAB **SETMRG** set left margin **SETHOM** set home position clear all column tabs **CLRTAB** (*)APLON apl on apl off (*)APLOFF treat input as ascii (*)APLEND (*)PCON xon/xoff on xon/xoff off (*)PCOFF disconnect (suspend) DISC new terminal type (*)INIT alternate keyboard dvorak (*)ALTK **FLINP** flush input

ERASE

erase last character

MAP3270(5) MAP3270(5)

```
WERASE
                      erase last word
  FERASE
                      erase field
 SYNCH
                      we are in synch with the user
                      reset key-unlock keyboard
  RESET
  MASTER_RESET
                      reset, unlock and redisplay
(*)XOFF
                      please hold output
(*)XON
                      please give me output
  ESCAPE
                      enter telnet command mode
  WORDTAB
                      tab to beginning of next word
  WORDBACKTAB
                      tab to beginning of current/last word
  WORDEND
                      tab to end of current/next word
  FIELDEND
                      tab to last non-blank of current/next
                      unprotected (writable) field.
 PA<sub>1</sub>
                      program attention 1
 PA2
                      program attention 2
 PA3
                      program attention 3
 CLEAR
                      local clear of the 3270 screen
 TREQ
                      test request
 ENTER
                      enter key
 PFK1
                      program function key 1
 PFK2
                      program function key 2
 etc.
                      etc.
 PFK36
                      program function key 36
```

A SAMPLE ENTRY

The following entry is used by tn3270(1) when unable to locate a reasonable version in the user's environment and in /etc/map3270:

```
# actual name comes from TERM variable
name {
clear = '^z';
flinp = '^x';
enter = '^m';
delete = '^d | '^?;
                        # note that '^?' is delete (rubout)
synch = '^r';
reshow = '^v';
eeof = '^e';
tab = '^i';
btab = '^b':
nl = '^n';
left = '^h';
right = '^1;
up = '^k';
down = '^j';
einp = '^w'
reset = '^t:
xoff = '^s';
xon = '^q';
escape = '^c;
ferase = '^u';
insrt = \langle E';
# program attention keys
```

MAP3270(5)

```
pa1 = '^p1'; pa2 = '^p2'; pa3 = '^p3'; # program function keys pfk1 = '\E1'; pfk2 = '\E2'; pfk3 = '\E3'; pfk4 = '\E4'; pfk5 = '\E5'; pfk6 = '\E6'; pfk7 = '\E7'; pfk8 = '\E8'; pfk9 = '\E9'; pfk10 = '\E0'; pfk11 = '\E-'; pfk12 = '\E-'; pfk13 = '\E!'; pfk14 = '\E@'; pfk15 = '\E#'; pfk16 = '\E$'; pfk17 = '\E%'; pfk18 = '\E'; pfk19 = '\E&'; pfk20 = '\E*'; pfk21 = '\E('; pfk22 = '\E)'; pfk23 = '\E_'; pfk24 = '\E+'; }
```

IBM 3270 KEY DEFINITONS FOR AN ABOVE DEFINITION

The charts below show the proper keys to emulate each 3270 function when using the default key mapping supplied with tn3270(1) and mset(1).

Type of Key	IBM 3270 Key	Default Key(s)
Command Keys	Enter Clear	RETURN control-z
Cursor Movement Keys	New Line Tab Back Tab Cursor Left Cursor Right Cursor Up Cursor Down	control-n or Home control-i control-b control-h control-l control-k control-j or LINE FEED
Edit Control Keys	Delete Char Erase EOF Erase Input Insert Mode End Insert	control-d or RUB control-e control-w ESC Space ESC Space
Program Function Keys	PF1 PF2	ESC 1 ESC 2
	PF10 PF11 PF12 PF13 PF14	ESC0 ESC - ESC = ESC! ESC @
	 PF24	ESC +
Program Attention Keys	PA1 PA2 PA3	control-p 1 control-p 2 control-p 3
Local Control Keys	Reset After Error Purge Input Buffer Keyboard Unlock Redisplay Screen	control-r control-x control-t control-v
Other Keys	Erase current field	control-u

MAP3270(5) MAP3270(5)

FILES

/etc/map3270

SEE ALSO

tn3270(1), mset(1)

Yale ASCII Terminal Communication System II Program Description/Operator's Manual (IBM SB30-1911)

BUGS

Tn3270 doesn't yet understand how to process all the functions available in map3270; when such a function is requested tn3270 will beep at you.

The definition of "word" (for "word erase", "word tab") should be a run-time option. Currently it is defined as the kernel tty driver defines it (strings of non-whitespace); more than one person would rather use the "vi" definition (strings of specials, strings of alphanumeric).

PRINTER3812(5) PRINTER3812(5)

NAME

printer3812 - IBM 3812 Pageprinter status information

SYNOPSIS

#include < printer3812.h >

DESCRIPTION

Printer3812.h describes the status information that is read from the printer. It also contains the print_3812_flags data structure that is used to indicate which 3812 messages are to be passed to the filter.

```
* Messages from Printer
#define POWER ON
                             0x14
                                    /* Power on reset status
                                                                  */
#define PAGE DONE
                                    /* Page in exit tray
                                                                  */
                             0x10
                                                                  */
#define INTERVENTION
                             0x11
                                    /* Intervention required
                                    /* Intervention cleared
                                                                  */
#define I CLEARED
                             0x15
#define COMMAND ERR
                             0x12
                                    /* Command exception status
                                                                  */
#define PRINTER ERR
                             0x13
                                    /* Printer Failure
                                                                  */
/*
 * Intervention Required types:
#define PAPER JAM
                             0x01
                                    /* Paper jam
#define TONER LOW
                             0x02
                                    /* Toner supply low
                                                                  */
#define EXIT FULL
                                    /* Exit tray full
                             0x03
#define OUT_OF_PAPER
                             0x04
                                    /* Paper supply empty
#define STOP KEY
                                    /* Stop key depressed
                                                                  */
                             0x05
#define CANCEL KEY
                                    /* Cancel key depressed
                                                                  */
                             0x06
#define PMP STOP
                                                                  */
                             0x07
                                    /* PMP STOP command
                                    /* Cover open
#define COVER OPEN
                             80x0
                                                                  */
                                    /* Job offset malfunction
#define OFFSET_ERR
                             0x09
                                                                  */
#define MEMORY ERR
                             0x0A
                                    /* Page Map memory error
                                                                  */
 * Intervention Cleared types:
 */
#define CLR_PAPER_JAM
                             0x01
                                    /* Paper jam cleared
#define CLR_TONER_LOW
                                    /* Toner supply low cleared
                                                                  */
                             0x02
#define CLR EXIT FULL
                                    /* Exit tray full cleared
                                                                  */
                             0x03
                                                                  */
#define CLR NO PAPER
                             0x04
                                    /* Paper supply replenished
#define START KEY
                                    /* Start key depressed
                                                                  */
                             0x05
#define CLR_COVER_OPEN 0x08
                                    /* Cover closed
 * Command exception status types:
#define BAD PMP CMD
                             0x01
                                    /* Invalid PMP command code */
#define BAD PMP PARAM
                             0x02
                                    /* Invalid PMP parameter
                                                                  */
#define OUT OF PAGE PAT 0x03
                                    /* Pattern out of page
                                                                  */
#define OUT OF PAGE VEC 0x04
                                                                  */
                                    /* Vector out of page
#define OUT OF STORAGE 0x05
                                    /* Insufficient storage
                                                                  */
#define UNDEFINED LIB
                             0x06
                                    /* Undefined library file
                                                                  */
#define EXCEED NEST LIM 0x07
                                    /* Macro nesting limit exceed
                                                                  */
static
char *irstatus[] = {
                      "Unknown",
```

PRINTER3812(5) PRINTER3812(5)

```
"Paper jam",
                       "Toner supply is low",
                       "Exit tray is full",
                       "Out of paper",
                       "Stop key was depressed",
                       "Cancel key was depressed",
                       "Printer stopped via PMP STOP command",
                       "Cover open",
                       "Job offset malfunction",
                       "Page map memory error",
#define IRSTATUS CNT (sizeof irstatus/sizeof irstatus[0])
static
char *ircleared[] = {
                       "Invalid status",
                       "Paper jam cleared",
                       "Toner supply replenished",
                       "Exit tray cleared",
                       "Paper supply replenished",
                       "Start key",
                       "Invalid status(6)",
                       "Invalid status(7)",
                       "Cover closed".
#define IRCLEARED CNT (sizeof ircleared/sizeof ircleared[0])
static
char *cmd excp[] = {
                       "Invalid status",
                       "Bad PMP command"
                       "Bad PMP parameter",
                       "Out of page pattern",
                       "Out of page vector",
                       "Out of storage in printer",
                       "Undefined library file",
                       "Macro nesting limit exceeded"
#define CMD EXCP CNT (size of cmd excp/size of cmd excp[0])
 * The following structure (print 3812 flags) indicates what error the calling
* program wants to know about. There is one field (short) for each error
 * message defined for the IBM 3812 Pageprinter. If the field is 0,
* then ignore; if 1, then return the message to filter.
#define IBM3812 PMP DATA 1
#define IBM3812 ASCII DATA 2
#define IBM3812 IGNORE ERR MSG 0
#define IBM3812 RETURN ERR MSG 1
typedef struct {
   short type;
   short power on;
   short page done;
   short intervention[IRSTATUS CNT];
```

PRINTER3812(5) PRINTER3812(5)

```
short i_cleared[IRCLEARED_CNT];
short command_err[CMD_EXCP_CNT];
short printer_err;
long sequence_id;
char username[30];
char hostname[30];
} print_3812_flags;

SEE ALSO
ibm3812pp(8), ppt(8)
IBM 3812 Pageprinter Programming Reference, S544-3268
```

RC.CONFIG(5) RC.CONFIG(5)

NAME

rc.config - configuration file for startup scripts

SYNOPSIS

. /etc/rc.config

DESCRIPTION

Rc.config is the configuration file that Bourne-shell startup scripts use to determine how to automatically start up the system. Currently the files that use rc.config are: /etc/rc, /etc/rc.local, /etc/viced, and, on the minimal system (see system_type, below), /etc/init. Note that shell scripts that are written to /etc/rc.config should not assume it is always there, since that assumption is not made by the system startup scripts. Also note that rc.config does not export any variables to the environment. This is especially important since on the minimal system, /etc/init would be affected. The following is a list of all of the parameters, with their defaults, that are specified in rc.config.

hostname

The hostname of the machine. It is used by /etc/rc.local for setting the hostname using /bin/hostname. On the minimal system, it is also used by /etc/init for starting up the network with /etc/ifconfig. The default value for the hostname parameter is master.

network

The network that the machine will use. For example, in order to use the token-ring, you should define: 'network = lan0'. The default value for network is un0.

net_flags

Any additional flags you wish to have passed on to /etc/ifconfig by /etc/rc.local (/etc/init on the minimal system) should be placed in this parameter. The default value for net flags is the null string.

system_type

The type of system that is loaded on the machine. There are three types of systems: full, which means both the root and usr file systems are local to the machine; reduced, which means the root partition is local to the machine, and the usr file system is obtained through the Andrew File System; and minimal, which means that only a bare-minimum root partition is local to the machine. The majority of the root partition is obtained via RVD, and, like the reduced system, the usr file system is obtained via the Andrew File System. The default value for system type is full.

core_directory

The directory where /etc/savecore places system dumps, if any. The default value for core_directory is /usr/tmp.

use_timed

This parameter is used to determine whether the machine is to start up /etc/timed during startup. If it is set to yes, /etc/timed will be started. The default value is no.

startvenus

If the machine is a client of the Andrew File System, /etc/viced will call this program to start up the Andrew File System daemon: venus(8V). The default value for start-venus is /etc/startvenus.

venus

If the machine is a client of the Andrew File System, startvenus(8V) will start this program if invoked by /etc/viced. The default value for venus is /etc/venus2.

venus_cache The location of the cache directory that venus(8V) will use. The default value for venus_cache is /usr/venuscache.

andrew servers

A comma-separated list of Andrew File System servers that venus(8V) will connect to. The default value for andrew servers is vice1.

venus_device The device that venus(8V) will use to route file system requests through. The default value for venus device is |dev|fs0.

andrew_dir The directory that startvenus(8V), if invoked with /etc/viced, will attempt to mount venus_device on. The default directory is /andrew.

RC.CONFIG(5) RC.CONFIG(5)

SEE ALSO

rc(8), viced(8V), venus(8V), startvenus(8V), ifconfig(8C), timed(8), savecore(8) "Installing and Operating IBM/4.3"

RVDDB(5)

NAME

rvddb - Remote Virtual Disk (RVD) server configuration table

DESCRIPTION

Rvddb contains a sequence of operation requests used to intialize the RVD server. These operations are typically add_physical and add_virtual types, which define the partitions to be used by the server, and how these partitions are allotted to the RVD packs, respectively. This configuration information is loaded into the server at boot-time (see rvdsend(8)). For a complete description of the protocol, refer to the document referenced below.

Other operations may be added; however, they will be deleted by vddb (see vddb(8)) when it is used on this file. For security reasons, the file is readable only by root.

FILES

/etc/rvd/rvddb

SEE ALSO

rvdsend(8), rvdsrv(8), vddb(8)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

RVDTAB(5)

NAME

/etc/rvd/rvdtab - information about client Remote Virtual Disks (RVDs)

SYNOPSIS DESCRIPTION

/etc/rvd/rvdtab file contains information that is used by the up, down, and rvdflush utilities to manipulate remote virtual disks. These utilities only read /etc/rvd/rvdtab; they do not write to it. The system administrator is responsible for maintaining the file.

The following is a typical /etc/rvd/rvdtab file:

```
vsusr 0
              agamemnon r /urvd
                                       # default /usr
  vsusr 0
              andromache r /urvd
                                      # backup /usr
                                      # default sys
  vssvs 1
              agamemnon r /srvd
  vssys 1
                                      # backup sys
              andromache r /srvd
                                    # system sources
% source 2
             priam r /src
  appdev 3
             jason r /usr/mac amyscam # macsyma disk
        4 helen xr /mit/me/rdir
                                  # my locker
```

Each line in the file contains a single rvd pack entry. Each entry has this form:

```
{* | %} pack drive host modes dir [ passwd ] [ # comment ]
```

You can use spaces or tabs to separate the fields. Comments can be placed on lines by themselves, and you can have blank lines as well.

The pack field specifies the rvd's pack name. This is the pack's name on the rvd server.

The *drive* field tells the *up* utility which of the workstation's virtual disk drives to use when spinning up the pack. Under the current configuration, each workstation has ten virtual disks, numbered 0 through 9.

The host field specifies the pack's server host. up, down, and rvdflush use this field to determine where to make spinup and spindown requests.

The modes field specifies the mode or modes in which the pack can be spun up. r specifies readonly mode; x specifies exclusive mode. If you list both modes in the mode field (see the seventh entry in the example above), up will by default spin up the disk in the first mode listed. up has options that let you get around the default.

The dir field tells up where to mount the spun-up pack. This field must contain an absolute pathname of a directory, starting with the "/" character.

The optional passwd field (see line six in the example) is now obsolete. You should no longer specify pack passwords in the table. If a pack requires a password, the up command will prompt you for it. up also has a -p option that lets you specify pack passwords in the up command line. This option should be used only by programmers writing shell scripts that invoke the up command. Public library packs, such as vssys and vsusr do not require passwords.

Entries may be flagged by a '*' or '%' to indicate that they are "default" RVD packs (eg, "up -d" applies to them). The difference is '*' indicates a required pack, and the user may specify that up(1) repeatedly attempt to spinup as many as possible in a round-robin fashion until all '*' packs have been spun up or cannot be spun up because the target drive is already in use. Typical usage for the '*' flag is to mark all "/urvd" and "/srvd" entries in /etc/rvd/rvdtab. The '%' flag marks an entry as a "default pack", but should the spinup fail, no further attempts are made ("a desirable disk, but not "absolutely" necessary").

Note that the order of the file's entries is important because up(1), down(1), and rvdflush(8) read |etc/rvd/rvdtab| from top to bottom as they process disk packs. For instance, at boot time, when up attempts to spin up a vsusr rvd pack, the utility will try to spinup the first vsusr entry in the file. If the spinup fails, up will move to the next entry in the file and try to spinup that pack.

RVDTAB(5)

The proper way to read records from |etc/rvd/rvdtab| is to use the library of routines written for up(1) and down(1).

FILES

/etc/rvd/rvdtab

remote virtual disk table

SEE ALSO

up(1), down(1), rvdflush(8)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

BUGS

Flagging a pack with '*' that requires a password will produce unrecoverable errors, particularly during reboot if /etc/rc.local calls up(1) to spinup any RVD disks.

IBM/4.3 Manual Pages

Section 8. Maintenance Commands and Procedures

This section contains information related to system operation and maintenance. Man pages found in IBM/4.3, but not in 4.3BSD, are marked with an asterisk (*). Man pages marked with a section symbol (§) are at a level earlier than 4.3BSD.

•	intro	•	restore
•	aedtest*	•	restore.tape*
•	afpacode(8R)*	•	rvdchlog*
•	badsect	•	rvdcopy*
•	config§	•	rvddown*
•	crash(8R)	•	rvdexch*
•	cvt3812*	•	rvdflush*
•	cvtsym*	•	rvdgetm*
•	debug*	•	rvdhosts*
•	diskpart*	•	rvdlog*
•	fdformat(8R)*	•	rvdsend*
•	fdisk*	•	rvdsetm*
•	flcopy(8R)*	•	rvdshow*
•	format(8R)	•	rvdshut*
•	halt	•	rvdsrv*
•	ibm3812pp*	•	sautil(8R)*
•	ifconfig(8C)	•	savervd*
•	init	•	scsiformat(8C)*
•	landump(8R)*	•	sendapar*
•	lpfilter(8R)*	•	setid
•	makedev	•	setscreen*
•	makesym*	•	spinup*
•	minidisk(8R)*	•	syscall*
•	newfs	•	syslogd
•	newvd*	•	tailor*
•	omerge*	•	tbuffer*
•	ppt*	•	vdabort*
•	pstat	•	vddb*
•	ptfinstall*	•	vdstats*

reboot

width3812*

INTRO(8)

NAME

intro - introduction to system maintenance and operation commands

DESCRIPTION

This section contains information related to system operation and maintenance. In particular, commands used to create new file systems, newfs and mkfs, and verify the integrity of the file systems, fsck, icheck, dcheck, and ncheck, are described here.

LIST OF PROGRAMS

Program	Appears on Page	Description
ac	ac.8	login accounting
accton	sa.8	system accounting
adduser	adduser.8	procedure for adding new users
aedtest	aedtest.8	IBM Academic Information Systems experimental display
		self-tests
analyze	analyze.8	virtual postmortem crash analyzer
badsect	badsect.8	create files to contain bad sectors
bugfiler	bugfiler.8	file bug reports in folders automatically
catman	catman.8	create the cat files for the manual
chown	chown.8	change owner
clri	clri.8	clear i-node
comsat	comsat.8c	biff server
config	config.8	build system configuration files
crash	crash.8r	what happens when the system crashes
cron	cron.8	clock daemon
cvt3812	cvt3812.8	convert IBM 3820 and IBM 3800 fonts for use with the
		IBM 3812 Pageprinter
cvtsym	cvtsym.8	convert symbol table
dcheck	dcheck.8	file system directory consistency check
debug	debug.8	debugger for the IBM RT PC
diskpart	diskpart.8	calculate default disk partition sizes
dmesg	dmesg.8	collect system diagnostic messages to form error log
drtest	drtest.8	standalone disk test program
dump	dump.8	incremental file system dump
dumpfs	dumpfs.8	dump file system information
edquota	edquota.8	edit user quotas
fastboot	fastboot.8	reboot/halt the system without checking the disks
fasthalt	fastboot.8	reboot/halt the system without checking the disks
fdformat	fdformat.8r	format diskettes
flcopy	flcopy.8r	copier for diskette
format	format.8r	how to format disk packs
fsck	fsck.8	file system consistency check and interactive repair
ftpd	ftpd.8c	DARPA Internet File Transfer Protocol server
gettable	gettable.8c	get NIC format host tables from a host
getty	getty.8	set terminal mode
halt	halt.8	stop the processor
htable	htable.8	convert NIC standard format host tables
ibm3812pp	ibm3812pp.8	IBM 3812 Pageprinter server
icheck	icheck.8	file system storage consistency check
ifconfig	ifconfig.8c	configure network interface parameters
implog	implog.8c	IMP log interpreter
implogd	implogd.8c	IMP logger process
init	init.8	process control initialization
kgmon	kgmon.8	generate a dump of the operating system's profile buffers

INTRO(8) INTRO(8)

landump	landump.8r	dump IBM Token-Ring Personal Computer Adapter
lpfilter	lpfilter.8r	output filters for the IBM 4201 Proprinter
		and IBM 5152 Graphics Printer
lpc	lpc.8	line printer control program
lpd	lpd.8	line printer daemon
makedev	makedev.8	make system special files
makekey	makekey.8	generate encryption key
minidisk	minidisk.8r	minidisk maintenance utility
mkfs	mkfs.8	construct a file system
mklost + found	mklost + found.8	make a lost + found directory for fsck
mknod	mknod.8	build special file
mkproto	mkproto.8	construct a prototype file system
mount	mount.8	mount and dismount file system
ncheck	ncheck.8	generate names from i-numbers
newfs	newfs.8	construct a new file system
omerge	omerge.8	merge object files
pac	pac.8	printer/plotter accounting information
ppt	ppt.8	text filter for the IBM 3212 Pageprinter
pstat	pstat.8	print system facts
quot	quot.8	summarize file system ownership
quotacheck	quotacheck.8	file system quota consistency checker
quotaoff	quotaon.8	turn file system quotas on and off
quotaon	quotaon.8	turn file system quotas on and off
rc	rc.8	command script for auto-reboot and daemons
rdump	rdump.8c	file system dump across the network
reboot	reboot.8	UNIX bootstrapping procedures
renice	renice.8	alter priority of running processes
repquota	repquota.8	summarize quotas for a file system
restore	restore.8	incremental file system restore
rexecd	rexecd.8c	remote execution server
rlogind	rlogind.8c	remote login server
rmt	rmt.8c	remote magtape protocol module
route	route.8c	manually manipulate the routing tables
routed	routed.8c	network routing dacmon
rrestore	rrestore.8c	restore a file system dump across the network
rshd	rshd.8c	remote shell server
rvdchlog	rvdchlog.8	change logging level of Remote Virtual Disk (RVD) server
rvddown	rvddown.8	force spindown of a Remote Virtual Disk (RVD) pack
rvdexch	rvdexch.8	exchange names of two Remote Virtual Disk (RVD) packs
rvdflush	rvdflush.8	spindown client's Remote Virtual Disk (RVD) pack
rvdgetm	rvdgetm.8	get operations message from Remote Virtual Disk (RVD) server
rvdlog	rvdlog.8	cause Remote Virtual Disk (RVD) server to log statistics
rvdsend	rvdsend.8	send control stream to Remote Virtual Disk (RVD) server
rvdsetm	rvdsetm.8	set operations message on Remote Virtual Disk (RVD) server
rvdshow	rvdshow.8	show connections to Remote Virtual Disk (RVD) server
rvdshut	rvdshut.8	force shutdown of Remote Virtual Disk (RVD) server
rvdsrv	rvdsrv.8	Remote Virtual Disk (RVD) server daemon
rwhod	rwhod.8c	system status server
sa	sa.8	system accounting
sautil	sautil.8r	standalone utility package
savecore	savecore.8	save a core dump of the operating system
savervd	savervd.8	back up Remote Virtual Disk (RVD) packs to tape
sendmail	sendmail.8	send mail over the internet

INTRO(8) INTRO(8)

sticky swapon sync syslog telnetd tftpd trpt tunefs umount update uuclean uusnap vddb vdstats vipw width 3812	swapon.8 sync.8 syslog.8 telnetd.8c tftpd.8c trpt.8c tunefs.8 mount.8 update.8 uuclean.8c uusnap.8c vddb.8 vdstats.8 vipw.8	specify additional device for paging and swapping update the super block log systems messages DARPA TELNET protocol server DARPA Trivial File Transfer Protocol server transliterate protocol trace tune up an existing file system mount and dismount file system periodically update the super block uucp spool directory clean-up show snapshot of the UUCP system Remote Virtual Disk (RVD) database manager list client Remote Virtual Disk (RVD) statistics edit the password file
width3812	width3812.8	build width tables for IBM 3812 Pageprinter fonts

AEDTEST(8) AEDTEST(8)

NAME

aedtest - IBM Academic Information Systems experimental display self-tests

SYNOPSIS

aedtest

DESCRIPTION

Aedtest invokes a suite of test routines to check the various components of the IBM Academic Information Systems experimental display unit. Aedtest should be called following system installation to verify correct operation; thereafter, it may be called whenever desired.

Since aedtest takes control of the display, it is best run as a foreground task. Aedtest writes its findings in the files listed below, then returns control of the display to the user.

Error indications should be reported to the user's service support activity. Occasionally, a "timeout" may be reported as an error. This does not necessarily mean a defect has been found.

NOTE

Aedtest applies only to the IBM Academic Information Systems experimental display.

FILES

diagnostic.lst aederrs.lst results of the aedtest session listing of errors encountered

BUGS

Will sometimes report a "timeout" in the results files. This can happen even with reliable equipment, and does not necessarily indicate a display problem.

AFPACODE(8R) AFPACODE(8R)

NAME

afpacode - load, test, and bring online the Advanced Floating Point Accelerator

SYNOPSIS

afpacode [-ucode filename] [-afpamem filename] [-rc]

DESCRIPTION

Afpacode is used to load microcode into the Advanced Floating Point Accelerator (AFPA). Until the operational microcode is loaded, the AFPA is considered offline by the system. After the AFPA has been loaded (and the load verified by reading back the contents of the microcode store), the AFPA is enabled (by disabling control store accesses), and (after a test) is brought online to the system.

The microcode description file is nominally standard input but may be specified with the -ucode flag. This file contains the actual contents to be loaded into the control store, plus the number of register sets which must be reserved (by the operating system) for use by the microcode, plus the initial contents of (some of) the (reserved) register sets. The format of the microcode description file is described below in the section MICROCODE DESCRIPTION FILE.

- ucode filename

Specifies a file which contains the microcode description file. The default is standard input.

- afpamem filename

Specifies the special memory file into which the microcode will be loaded (and read back from). If not specified, this defaults to /dev/afpamem.

-rc This flag, typically used to inform afpacode that it is being driven from /etc/rc.local, changes the message output of afpacode. Systems with no AFPA installed will see no messages from afpacode; systems in which the AFPA (through whatever mechanism) has already been loaded will see no messages from afpacode; and afpacode will print a message indicating that it is loading the AFPA in the case that the AFPA is on the system and in need of being loaded.

- offset value

Specify an offset into the control store to start the load at. This is used only for debugging the AFPA.

- interactive

After parsing the microcode description file (unless the -noucode option is specified), enter an interactive debugging mode. The commands available in this mode are listed and described in the section "INTERACTIVE MODE" below.

- dontstart

Don't attempt to bring the AFPA online. Don't disable control store accesses. Additionally, don't complain if the supplied microcode is not the correct length for the AFPA.

verbose

Produce more information about what afpacode is doing, and a longer run of messages when failures occur.

dontloaducode

Don't actually load the microcode into the AFPA.

— noucode

Do not attempt to parse the microcode description file. This should be used with the -interactive option, and then only if it is desired to test the AFPA code without loading the operational microcode.

query

The only actions of afpacode are to parse the arguments, query the system to find out

AFPACODE(8R) AFPACODE(8R)

what hardware is on the system, and to print out the state of the AFPA hardware.

- debug

Enable certain debugging capabilities. These are for debugging some of the parsing functions of afpacode.

INTERACTIVE MODE

Interactive mode is entered when the user has entered the -interactive option. Interactive mode is used to try to determine a failure mode in the AFPA control store. The commands in interactive mode are:

pattern pattern

Specify a pattern to be used for writing (and checking) the control store. The available patterns are:

ones All bits turned on.

zeroes All bits turned off.

55 Alternating bits.

aa Alternating bits.

address Each location written with its address.

random Each location has a random value.

ucode Use the contents of the microcode array Parsed from the microcode description file.

length How many microcode words to write, read, or check. Note that a microcode word is 64 bits of data.

offset Where in the control store to begin writing, reading, or checking.

write Write the pattern space into the control store. This uses the current values of *length* and offset.

read Read from the control store into the check array. This uses the current values of length and offset.

check Check the check array against the pattern space. This uses the current values of length and offset.

wrc Does a write, followed by a read, followed by a check. This uses the current values of length and offset.

help

? Print out some help information.

print Print out the contents of the of the check array. This uses the current values of length and offset.

quit Terminate afpacode.

MICROCODE DESCRIPTION FILE

Comments are indicated by a hash mark (#). They continue to the end of the current line. There are three statements recognized: reserved, registerset, and microcode:

reserved

Informs the system of how many register sets in the AFPA are to be reserved for use by the AFPA microcode.

AFPACODE(8R) AFPACODE(8R)

registerset

Gives the initial contents of a specific register set (more than one registerset statement may be specified).

microcode

Gives the contents of the control store.

The format of the statement is as follows:

```
reserved = number;
```

registerset[which] = {'00000000'XB, ...};
"which" must be in the range [0-31]. There must be 64 values defined for each register set.

microcode = {'39800240010000A0'XC, ... };

There must be 4096 values defined for the control store.

EXAMPLE

/usr/ibm/afpacode -ucode /usr/ibm/lib/afpa.ucode -rc

NOTE

Afpacode is not supported on the IBM 6152 Academic System.

BADSECT(8)

BADSECT(8)

NAME

badsect - create files to contain bad sectors

SYNOPSIS

/etc/badsect bbdir sector ...

DESCRIPTION

Badsect makes a file to contain a bad sector. Normally, bad sectors are made inaccessible by the standard formatter, which provides a forwarding table for bad sectors to the driver; see disk(4) and format(8R) for details. If a driver supports the bad-blocking standard it is much preferable to use that method to isolate bad blocks, since the bad-block forwarding makes the pack appear perfect, and such packs can then be copied with dd(1). The technique used by this program is also less general than bad-block forwarding, as badsect can't make amends for bad blocks in the i-list of file systems or in swap areas.

Badsect is used on a quiet file system in the following way: mount the file system, and change to its root directory. Make a directory BAD there. Run badsect giving as argument the BAD directory followed by all the bad sectors you wish to add. (The sector numbers must be relative to the beginning of the file system, but this does not pose a problem, because the system reports relative sector numbers in its console error messages.) Next, change back to the root directory, unmount the file system and run fsck(8) on the file system. The bad sectors should show up in two files or in the bad sector files and the free list. Have fsck remove files containing the offending bad sectors, but do not have it remove the BAD/nnnnn files. This would leave the bad sectors only in the BAD files.

Badsect works by giving the specified sector numbers in a mknod(2) system call, creating an illegal file, the first block address of which is the block containing the bad sector(s), and the name of which is the bad-sector number. When the file is discovered by fsck, it will ask, hold bad block? A positive response will cause fsck to convert the inode to a regular file containing the bad block.

SEE ALSO

disk(4), format(8R), fsck(8)

DIAGNOSTICS

Badsect refuses to attach a block that resides in a critical area or is out of range of the file system. A warning is issued if the block is already in use.

BUGS

If more than one sector comprising a file system fragment is bad, specify only one of them to badsect, as the blocks in the bad-sector files cover all the sectors in a file-system fragment.

CONFIG(8) CONFIG(8)

NAME

config - build system configuration files

SYNOPSIS

/etc/config [- p] system_name

DESCRIPTION

Config builds a set of system-configuration files from a short file that describes the system being configured. Config also takes as input a file that tells it which files are needed to generate a system. This can be augmented by a configuration-specific set of files that gives alternate files for a specific machine (see "Files", below). If the $-\mathbf{p}$ option is supplied, config will configure a system for profiling; see kgmon(8) and gprof(1).

Config should be run from the conf subdirectory of the system source (usually, /sys/conf). Config assumes that there is already a directory ../system_name created and places all its output files there. The output of config consists of several files: ioconf.c contains a description of the I/O devices attached to the system. Makefile is used by make(1) in building the system; header files contain the number of various devices to be compiled into the system; and swap-configuration files contain definitions for the disk areas used for swapping, the root file system, argument processing and system dumps.

After running config, it is necessary to run "make depend" in the directory where the new makefile was created. Config reminds you of this when it completes.

If config issues error messages, fix the problems and try again. Do not compile a system that had configuration errors.

FILES

/sys/conf/makefile.ca generic makefile for the IBM RT PC

/sys/conf/files list of common files from which system is built

/sys/conf/files.ca list of IBM RT PC-specific files

/sys/conf/devices.ca name of major device mapping file for the IBM RT PC

/sys/conf/files.ERNIE list of files specific to ERNIE system

SEE ALSO

"Building IBM/4.3 Systems with Config" in Volume II, Supplementary Documents The "Synopsis" portion of each device in section 4

BUGS

The line numbers reported in error messages are usually off by one.

CRASH(8R) CRASH(8R)

NAME

crash - what happens when the system crashes

DESCRIPTION

When the system crashes voluntarily, it prints a message of the form

panic: why i gave up the ghost

on the console, takes a dump on a mass-storage peripheral, and then invokes an automatic reboot procedure as described in reboot(8). If the kernel was compiled with the debugger, the debugger is entered before the dump. A "go iar + 2" continues the dump and reboots. Unless some unexpected inconsistency is encountered in the state of the file systems due to hardware or software failure, the system will then resume multi-user operation.

The system has several internal-consistency checks; if one of these fails, the system panics, giving a brief message indicating which one failed.

The most common cause of system failures is hardware failure, which shows itself in different ways. The messages you are likely to encounter, along with some hints as to cause, are given below. (Left unstated in all cases is the possibility that hardware or software error produced the message in some unexpected way.)

I/O err in push

hard I/O err in swap

The system encountered an error trying to write to the paging device or in reading critical information from a disk drive. Fix the disk if it is broken or unreliable.

timeout table overflow

This really shouldn't be a panic, but until the data structure involved has been fixed, running out of entries causes a crash. If this happens, make the timeout table bigger.

trap

kernel trap not tlb

kernel trap

These indicate either a serious bug in the system or, more often, a glitch or failing hardware.

init died

The system-initialization process exited, blocking additional users from logging in. Rebooting is the only fix, so the system just does it right away.

When the system crashes, it writes, or at least attempts to write, an image of memory into the back end of the primary swap area. After the system is rebooted, the program savecore(8) runs and preserves a copy of this core image and the current system in a specified directory for later perusal (see savecore).

To analyze a dump, begin by running adb(1) with the -k flag on the core dump.

SEE ALSO

```
adb(1), analyze(8), reboot(8), savecore(8)
```

IBM RT PC Problem Determination Guide, SA23-1040, for more about hardware failures "Using ADB to Debug the Kernel", in the 4.3BSD UNIX System Manager's Manual

CVT3812(8)

NAME

cvt3812, cvt20to12, cvt00to12 - convert IBM 3820 and IBM 3800 fonts for use with the IBM 3812 Pageprinter

SYNOPSIS

```
cvt20to12 [ -c fontnameTBL ] [-M ] [-N] [-d ] font cvt00to12 [ -c fontnameTBL ] font
```

DESCRIPTION

The cvt20to12 program is used to convert fonts in the IBM 3820 format to a format for use with the IBM 3812 Pageprinter. The cvt00to12 program is used to convert fonts in the IBM 3800 format to a format for use with the 3812. The format of the files that are created is described in font3812(5).

Typographic fonts for the IBM 3812 Pageprinter are available from IBM as separate products. These fonts are shipped from IBM on diskettes in the IBM 3820 format. Once these font files have been loaded using the *dosread(1)* command, they must be converted using *cvt20to12*. If you have fonts in the IBM 3800 format, they can be converted using *cvt00to12*.

These programs accept the following options:

$-\mathbf{c}$ fontnameTBL

FontnameTBL contains a mapping between the IBM name and the installation name for the font. A typical entry for this table is:

ss SONORAN SANS SERIF

This will use the name ss as the font name for any SONORAN SANS SERIF font. The default fontnameTBL is /usr/lib/font/dev3812/fonts/fontnameTBL.

- -M This option does not convert the font. It produces the name of the file to be created.
- -N This option does not convert the font but produces a list of the IBM name for each character in the font.
- -d This is the debugging option to print font pel patterns.

These programs accept one argument, which is the filename of the font to be converted. For example, the input file for Sonoran Sans Serif size 6 from the diskette is named c0a05560.pds. The output file name is constructed from the name in fontnameTBL. Two output files are generated. The raster patterns for the font are in ss.6.dat, and the index (by IBM character name) to the raster patterns is in ss.6.ndx. These files are read by the width38/2(8) program which constructs the width tables for use by text formatters.

FILES

/usr/lib/font/dev3812/fonts/fontnameTBL

default fontname table

SEE ALSO

font3812(5), width3812(8)

CVTSYM(8)

NAME

cvtsym - convert symbol table

SYNOPSIS

cvtsym [option] ... input output

DESCRIPTION

Cvtsym converts an a.out file in VAX format into the format acceptable to a machine with different byte-ordering. It takes an a.out file with the header and symbol table in VAX byte order and changes them into the proper format for the target machine. It does not handle relocation information, so it is not suitable for running on a .o file from a compiler or assembler.

The following switches are accepted:

- -p# sets the logical page size (required to process ZMAGIC -- demand-paged -- files properly).
- -D prints internal debugging information about what cvtsym is doing.
- removes the "text" section from the resulting object file. Normally this is specified with -d, which leaves just the symbol table.
- -d removes the "data" section from the resulting object file.
- -n generates an NMAGIC-format output file. The output file will probably not be correct if the input file was OMAGIC.
- -N generates an OMAGIC-format output file. This is most useful when one wants to generate a "portable" symbol table file from a ZMAGIC input file.
- -z generates a ZMAGIC-format output file. This is not usually a good idea.
- -s does not generate a swapped header at the start of the output file.

NAME

debug - debugger for the IBM RT PC

SYNOPSIS

 $\sqrt{\frac{|w|}{|w|}} = \sqrt{\frac{|w|}{|w|}} = \sqrt{\frac{$

DESCRIPTION

The IBM RT PC debugger is designed to be used in standalone mode -- when it is linked with the program being debugged -- or for online kernel debugging, in which case it is linked separately, then loaded with the kernel, and for post-mortem kernel debugging, in which case it is a separate program called *debug*.

The kernel version, however, sets some global variables to change the environment and provide the symbol table.

The options have the following meanings:

- -w Opens mem for writing to allow changes to be made to the mem contents.
- -d Internal debugging flag.
- -n Don't pause for certain commands to read response (useful when *debug* is initiated from shell scripts).
- -b# Turns on buffering of data from mem. # sets the buffer size. A negative buffer size turns buffering off.
- mem The file or device that corrsponds to a memory image. It defaults to /dev/mem. It can be in the format described in a.out(5).

vmunix Contains the kernel namelist and defaults to /vmunix.

Entering the Debugger

Kernels compiled with a debugger can access the debugger using the $\{FN_DEBUG\}$ function assigned to the $\langle PAUSE \rangle$ key. This function can be redefined to any key using the pf(1) command. (See pf and kbdemul(4).) The debugger can also be entered using the $\langle Ctrl \rangle$ $\langle Alt \rangle$ $\langle Scroll\ Lock \rangle$ sequence.

The debugger will be invoked during boot and after panics.

Standalone utilities compiled with the debugger can access the debugger with the "*" command described below, or by generating a level 0 interrupt from the keyboard. It is possible to generate a standalone program with a debugger by making it with a dbg suffix, as the standalone makefile has a rule for this suffix.

Debugger Commands

Commands may be given from either the debug command mode (after a DEBUG > prompt) or at the standalone command level (after S/A %) with a leading "*". Commands may be abbreviated, but take care to avoid ambiguity.

A "null" command is taken as the previous command (with default operands),

so it is possible to display consecutive segments of memory by typing display, followed by pressing the < Enter > key several times, or to single-step through a program by typing step and then pressing the < Enter > key several times. The commands are given in alphabetical order, except that the command with the shorter abbreviation is given first. The following commands are available:

prints out the current status. It prints the reason for the current entry to the debugger, along with the IAR, ICS and CS at the time of the interrupt, and a symbolic printout of the MCPCS. These are followed by the general registers and the System Control Registers (SCRs). Any "watchpoints" tripped (see "Break and Watchpoints", below) are also printed.

? [cmd] is a synonym for help.

/half [addr [count]]

opens the given address for halfword modification. Each location is opened, the current contents displayed, and a new value read to replace the previous contents. The new value may be a debugger expression. Negative and zero values for *count* have special meanings discussed below. A total of *count* locations will be opened (unless terminated by an *end* replacement value; see below for more details).

/byte [addr [count]]

opens the given address for byte modification.

/word [addr [count]]

opens the given address for word modification.

= [addr [format]]

prints out the given address in the given format. The format is a printf format string and defaults to %x. This is a quick way of printing out the contents of a given register (e.g. "= r1").

ascii [addr]

displays memory at the specified address in ASCII characters. One common use of this command is to display the kernel console message buffer by the command "ascii msgbufp!4+8".

break [addr]

sets a breakpoint at the given address or, if no address is specified, lists the current breakpoints.

call addr [arg ...]

calls the C routine at the given address with up to three provided arguments. This is done on the debugger stack if the debugger has control (**DEBUG** > prompt), otherwise the normal stack is used.

clear [addr]

clears the breakpoint or watchpoint at addr. If no addr is specified, all the current breakpoints and watchpoints are cleared.

cls clears the screen.

define symbol addr

defines (or redefines) the value of the symbol symbol to given address value addr.

display [addr [count]]

displays the contents of the given address in hexadecimal and ASCII. Count lines of output will be produced (with lines that are all zero-suppressed, except for the first). Addr defaults to the next address (from the previous command) and count defaults to 10. Since the default command is the previous command, one may use a display command to display memory and then press the < Enter > key to display following memory locations.

dump [addr [count] [incr]]

displays *count* words at the address *addr* in various formats. The increment value *incr* (which defaults to 1) specifies how many words to increase *addr* by.

go [addr]

resumes execution after an interrupt (trap). It inserts the breakpoints into the code and transfers control either to the IAR at the time of the interrupt or to addr if it was specified. The ICS will have whatever value it had at the time of the interrupt (unless changed with a scr 14 value command).

hatipt [page [count]]

displays count entries in the hatipt (Hash Anchor Table/Inverted Page Table), starting at

the entry for page. The location of the hatipt is taken from the appropriate MMU registers.

help [cmd]

prints out a list of available commands or, if *cmd* is specified, the syntax and a quick summary of the effects of the given command. If *cmd* is "*", summaries of all the commands are given.

ident [addr [count]]

prints out all the RCS \$Header\$ lines in the given program or from the given address onward.

inb [port [count]]

reads a byte from the PC I/O port port and displays it. If count is specified then the processes is repeated that many times with port incremented by 1 each time.

inhw [port [count]]

reads a half word (16 bits) from the PC I/O port port and displays it. If count is specified then the processes is repeated that many times with port incremented by 1 each time.

ior [addr [count]]

issues an IOR instruction for the given address and following locations, for a total of count locations.

iow addr value

issues an IOW instruction for the given address and value.

lookup [symbol]

looks up all symbols in the symbol table beginning with symbol and prints them. If symbol is not specified, the entire symbol table is printed.

move from count to

moves count bytes of memory from from to to.

option [n [count]]

sets internal debugger options. Currently, n can have the following values: "1" selects the option that can be used to trace internal debugger operation; "2" selects automatic status-updating during step commands; "3" scans for symbol definitions in the program text (note that hc does not produce such definitions); "4" displays condition codes symbolically. The optional second value, count, is taken as the power of two of the number of instructions to execute between displays.

outb [port [value]]

writes the byte value to the PC I/O port port.

outhw [port [value]]

writes the halfword value to the PC I/O port port.

reflects a debugger interrupt back to whomever owned it before the debugger took over the interrupt vector. The debugger saves the previous value of the interrupt vector and the reflect command can be used to pass the interrupt to that program. WARNING: since the debugger has executed an "LPS" instruction and accessed various status registers, a reflected program-check or machine-check interrupt may behave differently from a directly serviced one.

register [number [value]]

with no operands, this displays the contents of the general registers. With two operands, the specified register is changed to the given value.

scr [number [value]]

with no operands, this displays all the system control registers. With two operands, the specified system control register is given the specified value. Not all registers can be

changed and not all bits of some registers can be changed. A warning is printed when a system control register does not end up with the requested value.

ser [value]

with no operands, this prints out the Storage Exception Register (SER), including the segment registers and the RAM and ROS specification registers. This gives useful information after a page fault or some other problem detected by the storage register. if value is specified, then this is displayed symbolically as the contents of the SER.

show [addr]

shows the contents of the screen at the time the debugger was entered. Use the < Print Screen > key to print, then cls to clear the screen. On subsequent entry to the debugger, the monochrome-screen image is saved at a specified address. If no address is specified, the debugger displays the contents of the monochrome-screen save region, which contains the screen contents at the point at which the debugger gained control. The debugger then reads a single character from the keyboard so that the screen may be viewed. In kernel debug mode, the screen is automatically saved when using the monochrome screen. Once the screen save area has been specified, the debugger automatically restores the screen when a go or step command is issued. An addr of "0" tells the debugger not to save the screen. This is sometimes useful when single-stepping.

step [[addr] count]

executes count instructions and returns control to the debugger. If two parameters are specified, the first is taken as an address at which to start execution (e.g. the IAR is set to addr); count instructions are then executed. Note that step will force the CPU priority to 1 if it was at 0 so that level-zero interrupts can be used to regain control after executing a single instruction. Currently, breakpoints are NOT set during the execution of a step command; however, they are checked for by address comparison. So, unless breakpoints are set on the second instruction of a branch-and-execute, they will be detected.

symbol [addr [count]]

prints out the symbol at addr (or the closest preceding one within count bytes). Count defaults to 0x10000 (65536).

symtab [addr]

notifies the debugger that a symbol table in the appropriate format is at the given address. In kernel mode this is done automatically, before the debugger gets control, by the "rdb" interface.

trace [iar] [sp]

traces back the C call stack and prints the parameters for each routine. *Iar* and *sp* may be specified instead of the IAR and SP at the time that the debugger was entered. This is particularly useful when the debugger has been entered after a kernel trap panic, where it useful to provide the IAR and SP from the kernel trap message.

unasm [addr [count]]

disassembles the given address and the following ones, up to *count* lines of output. If a symbol table is available, the symbolic address of the instruction is printed to the right of the disassembled code.

vector [value]

if no value is specified, the interrupt vectors for interrupt level 0, program check, machine check and the debugger's simulated interrupt vector at 0x1A0 are reset to the debugger. If value is specified, it defines a mask that shows which vectors to initialize. The interpretation of value is that used when requesting interrupts via the IRB. (For example, 8000 will reset the interrupt level 0 vector).

version prints out the current revision level of the debugger.

watch [addr]

sets a watchpoint at the given address or, if no address is specified, lists the current watch-points.

Break and Watchpoints

A breakpoint is a "TGTE R1,R1" instruction that is inserted in place of the actual instruction when executing in "non-stepped" mode.

The debugger can handle "static" breakpoints inserted into the source code in much the same way that it handles its own breakpoints.

Watchpoints are location values that are monitored whenever the debugger has control. Since the debugger has control after almost every instruction when in single-step mode, it is possible to locate the exact instruction that modifies a memory location by setting a watchpoint on that location. Only a fullword address can be a watchpoint.

It is possible to watch a general register, since general registers are saved in locore at location 0x300. Doing a watch 300 watches the value of R0.

Expressions

All commands accept several possible operand types:

- a hex number (of the form of 0xnnnnnnnn); e.g. 0x123b.
- a decimal number (of the form 0dnnnnnn); e.g. 0d12. Also, a number with a leading sign is taken as a decimal number.
- a symbolic address (if a symbol table is available); e.g. "main".
- a number without an explicit base. Depending on context, the value will be taken as either decimal or hexadecimal. Normally a number representing an address (such as the first operand in a command) will be taken as hexadecimal, while a number appearing as a count (such as the second operand) will be taken as decimal.
- the contents of a register (registers are specified by "r#" where "#" is in the range 0 < # < 15. (e.g. "r0"). Note that the contents of the register are used, rather than the "address" of the register. For example, if one has set a breakpoint at the start of a C procedure, then when that breakpoint trips, it is possible to set a breakpoint at the return address (which is in r15) by a break r15 command.

These operands may be combined with "+" and "-" operators in the usual ways. Spaces are not allowed in an expression.

Virtual Addresses

Any of the operand values may be preceded with the "=" monadic operator that takes the real address of a virtual address. Thus "=0" might have the value 0xfb000 if page zero of a user program is located there. If there is no physical page for a given virtual page, an appropriate message will be printed. This method of specifying a virtual address allows it to be used in any command that accepts an address.

Replacement Options

The "/" commands accept some optional values that control the interactive changing of memory locations:

expr is a normal address expression as given above.

end ends the "/" command. You may use "end" for the address of "end", if necessary.

backs up to the previous location in memory. The amounts backed up are 1, 2, or 4 bytes, as appropriate for the "/" command given.

space (or a null line) goes to the next location in memory.

Notes About the / Commands

Normally the "/" commands, which open memory locations for change, refer to the given location up to three times:

- 1 to validate the given address before using it.
- 2 to pick up the existing value for display.
- 3 (optionally) to deposit the new value entered.

In addition, the next location automatically opens after this one closes (subject to the value of count).

Sometimes, particularly with I/O addresses, this procedure can cause problems, since these hidden references change the expected behavior of the device being accessed. This can happen because some devices have only a small window; when accessing this window, one can get different internal registers with different references.

The solution is to use a special value in the *count* parameter of the "/" commands that cause abnormal behavior:

count = 0

here, the address validation reference is not done, so only the reference to read the old value and, possibly, the reference to write a new value are done.

count < 0

here, the only reference is to write a new value, if any, into the location. Because the old value is not picked up, only the address is printed.

In both of the above cases only the addressed location is opened. The following location is not opened automatically, because successive locations are often part of the same device.

Symbol Table

The debugger can use a symbol table to display addresses symbolically and to refer to addresses symbolically. The symbol table has a fixed format. Currently it has an 8-character name (with the leading "_" stripped), and a 4-byte value (address). This format was chosen because:

- it is compact
- it is the format used by RDB.
- it is in one piece (the a out format is in two pieces).
- it is easily processed.
- corrupting the symbol table will lead to a symbol not being found, but not to program checks inside the debugger.
- it can be truncated to fit a particular slot in memory without invalidating the non-truncated part.

The kernel version of the debugger has the symbol table automatically built by the kernel makefile.

Locore Locations Used by the Debugger

The debugger uses several "locore" locations for its operations.

- interrupt level zero vector. This is used in single instruction stepping.
- interrupt level one vector. This is normally reset in standalone mode but not kernel mode.
- interrupt level two vector. This is normally reset in standalone mode but not kernel mode.

- interrupt level three vector. This is normally reset in standalone mode but not kernel mode.
- interrupt level four vector. This is normally reset in standalone mode but not kernel mode.
- interrupt level five vector. This is normally reset in standalone mode but not kernel mode.
- interrupt level six vector. This is normally reset in standalone mode but not kernel mode.
- machine check vector. This intercepts machine checks.
- program check vector. This intercepts program checks.
- 190 SVC (supervisor call) vector. This is normally reset in standalone mode but not kernel mode.
- 1A0 A simulated vector used for debugger calls. This passes control to the debugger as if an interrupt had happened.
- is used when the debugger resumes execution after an interrupt. The debugger places a "lm r0,0x300" and a "lps t,vector" instruction here.

300...340

is used as a locore register save area by the debugger.

340...800

is used as an interrupt stack by the debugger as it cannot assume that the normal stack pointer is valid when an interrupt happens.

The debugger does not use locations 0...FF or locations 1B0...27F.

NOTES

It is possible to give commands to the debugger that will cause it to generate a program or machine check. This will cause recursive re-entry into the debugger, which will then lose track of its original entry reason. In that case, a *call* or *go addr* command should be used to transfer control to the original program at its entry point.

When using the *call* command from within the debugger after an interrupt, remember that the size-limited debugger stack is used. This is not a problem when the program's main entry point is called, since that establishes a new stack after a few instructions.

Single-stepping is carried out by requesting a level zero interrupt via the IRB, then doing an "LPS 1,...". If the instruction being executed suppresses interrupts (e.g. it is a LPS and sets the priority to zero or masks all interrupts) or clears the IRB IRQ 0 bit, then single-stepping will either be lost or suppressed until interrupts are again possible. This problem is handled by compiling the kernel with "-DLORDB" (specify option LORDB in the config(8) file), which causes the kernel not to go to priority level zero and not to clear the IRQ 0 bit. The intent of -DLORDB is to allow debugging of the kernel locore routines, particularly, the interrupt service code.

SEE ALSO

"Buidling Systems with Config" in Volume II, Supplementary Documents

DISKPART(8)

DISKPART(8)

NAME

diskpart - calculate default disk partition sizes

SYNOPSIS

/etc/diskpart [-p][-d] disk-type

DESCRIPTION

Diskpart calculates the disk partition sizes based on the default rules used at Berkeley. If the $-\mathbf{p}$ option is supplied, tables suitable for inclusion in a device driver are produced. If the $-\mathbf{d}$ option is supplied, an entry suitable for inclusion in the disk description file /etc/disktab is generated; compare with disktab(5). Space is reserved as follows: one cylinder at the front for configuration information and the bad-block-forwarding table; one cylinder at the back for diagnostic use; and enough cylinders at the back to hold 1000 replacement sectors. For more information, see disk(4).

The disk partition sizes are based on the total amount of space on the disk as given in the table below (all values are supplied in units of 512-byte sectors). The "c" partition is, by convention, used to access the entire physical disk, including the space reserved for the bad-sector-forwarding table. In normal operation, either the "g" partition is used, or the "d", "e", and "f" partitions are used. The "g" and "f" partitions vary in size, occupying whatever space remains after allocation of the fixed-sized partitions. If the disk is smaller than 20 megabytes, diskpart aborts with the message: disk too small, calculate by hand.

Partition	20-60 mb	61-205 mb	206-355 mb	356+ mb
a	15884	15884	15884	15884
ь	10032	33440	33440	66880
d	15884	15884	15884	15884
е	unused	55936	55936	307200
h	unused	unused	291346	291346

If an unknown disk type is specified, diskpart will prompt for the required disk geometry information.

NOTE

Since newfs(8) uses the partition data stored directly on the disk, diskpart(8) is obsolete in the System/2 environment. There is no default disk partitioning for each disk type.

SEE ALSO

disk(4), disktab(5), newfs(8)

BUGS

When using the -d flag, alternate disk names are not included in the output.

FDFORMAT(8R) FDFORMAT(8R)

NAME

fdformat - format diskettes

SYNOPSIS

/etc/fdformat [-h] special

DESCRIPTION

The fdformat program formats a diskette in the specified drive associated with the special file special. (Special is /dev/rfd0 for drive 0.) By default, the diskette is formatted for low-density (360K on the RT PC, or 720K on the Academic System 6152); a -h flag may be supplied to force high-density formatting (1.2m on the RT PC, or 1.4m on the Academic System 6152). 5 1/4" PC-DOS diskettes (360k) contain 40x2 tracks, each with 9 sectors (for a total of 720 sectors). 5 1/4" high-capacity diskettes (1.2m) contain 80x2 tracks, each with 15 sectors (for a total of 2400 sectors). 3 1/2" low-density diskettes contain 80 tracks, each with 9 sectors (for a total of 1440 sectors). 3 1/2" high-capacity diskettes contain 80 tracks, each with 19 sectors (for a total of 2880 sectors). The sector size is 512 bytes for both diskette types.

Before formatting a diskette, fdformat prompts for verification. This allows a user to abort the operation cleanly. Note that formatting a diskette destroys existing data.

Fdformat does not write DOS directory information on the disk.

FILES

/dev/rfd[01]

SEE ALSO

fd(4), flcopy(8R)

FDISK(8)

NAME

fdisk - boot record partition table maintenance utility

DESCRIPTION

Fdisk is a component of the sautil(8R) package of standalone utility programs. It is used to maintain boot record partition table entries. Fdisk displays and allows changes to be made to the partition table. It can be used to change the size and location of the 4.3 physical partition.

Fdisk Commands

There are a number of fdisk commands available:

create Creates a new 4.3 physical partition. There can be only one 4.3 physical partition per hard disk drive. The starting cylinder and partition length must be specified. (Both are value-checked.)

delete Deletes an entire physical partition. WARNING! Deleting a partition will effectively destroy any data on that partition. Be sure to back up anything you wish to save before doing this.

disk Specifies the current disk to be examined.

help Prints out a short description of each command.

list Prints (lists) the partition table for the current drive.

quit Quits fdisk and returns control to sautil(8R).

NOTE

Fdisk is not supported on the IBM RT PC.

SEE ALSO

boot(8), format(8R), minidisk(8R), newfs(8), sautil(8R)

Disk Operating System Version 3.30 (80X0667), First Edition (April 1987)

FLCOPY(8R) FLCOPY(8R)

NAME

flcopy - copier for diskettes

SYNOPSIS

/etc/flcopy [-f filename] [-h] [-r] [-tn]

DESCRIPTION

Flcopy copies a diskette (opened as /dev/rfd0) to a file created in the current directory, named "floppy", then prints the message "Change floppy, hit return when done". Flcopy then copies the local file back out to the diskette.

The -f option allows you to specify a file other than $\frac{dev}{rfd0}$.

The -h option causes floopy to open a file named "floppy" in the current directory and copy it to $\frac{dev}{fd0}$.

The $-\mathbf{r}$ option tells floopy to exit after copying the diskette into the file named "floopy" in the current directory.

The -t option causes only the first n tracks to participate in a copy.

DIAGNOSTICS

Prints "Not a floppy" if the device is not a diskette drive. Note that the raw device must be specified.

FILES

/dev/rfd0

floppy (in current directory)

SEE ALSO

fd(4), fdformat(8R)

FORMAT(8R) FORMAT(8R)

NAME

format - format hard disks

DESCRIPTION

Format is a standalone program used to format and check disks prior to constructing file systems. In addition to the formatting operation, format records any bad sectors encountered (see disk(4)). Formatting is performed one track at a time by writing the appropriate headers and a test pattern, then checking the sector by reading and verifying the pattern using the controller's Error Correcting Code (ECC) for error detection. A sector is marked bad if any media error is detected. After the entire disk has been formatted and checked, the total number of errors is reported, any bad sectors are marked, and bad-sector forwarding information is written to the disk. Format may be used on 40Mb and 70Mb disks supported by the hd driver.

The test pattern used during the media check may be selected from the following: 0xf00f (RH750 worst case), 0xec6d (media worst case), 0xa5a5 (alternating 1's and 0's) and 0x000. Normally the media worst-case pattern is used.

Format also has an option to perform an extended "severe burn-in", which makes 1 to 32 passes using different patterns. Under this option, any sectors with errors are marked bad. This test runs for many hours, depending on the disk size and repeat count.

Each time format is run, any new errors found will be added to the existing bad-block table based on errors encountered during formatting. The device driver, however, will always attempt to read an existing bad-sector table when the device is first opened. Thus, if a disk drive has lost its formatting information, error messages will be printed when the driver attempts to read the bad-sector table; these diagnostics should be ignored.

It is possible to erase the existing bad-block table by using the "change defaults" menu. This is useful when the reason for reformatting is that another disk's bad-block table was copied onto the disk to be formatted.

For every cylinder formatted, *format* prints a message indicating the cylinder currently being formatted. All the burn-in and multiple passes are made on each cylinder in turn, so the progress can be determined by the cylinder number printed.

Format uses the standard notation of the standalone I/O library in identifying a drive to be formatted. A drive is specified as zz(x,y), where zz refers to the controller type (hd), x is the unit number of the drive and y is the file system partition on drive x (this should be 2 for the entire IBM/4.3 partition of the disk). For example, hd(1,2) indicates that drive 1 on adapter 0 should be formatted; while hd(2,2) indicates drive 0 on adapter 1 should be formatted.

Format should be used prior to building file systems (with newfs(8)) to ensure that all sectors with uncorrectable media errors are remapped. If a drive develops uncorrectable defects after formatting, format should be used to add additional blocks by reformatting the cylinders containing the bad blocks. In this case, be sure to request that the original data be saved.

Format checks the keyboard after formatting each track so that it is possible to stop at a safe place by hitting the *enter* key, which will cause the program to ask the user whether formatting should be stopped. If the answer is yes, the normal exit processing will be done. Formatting can be resumed later by starting it again at the appropriate cylinder. A negative answer will continue formatting.

EXAMPLE

A sample run of format is shown below. Format was loaded from option 2, "format - format hard disk", of sautil(8R) on the IBM RT PC; it could also have been loaded from option 11, "convert - R70 ("hd70r") disk to an E70". Boldface means user input. As usual, "#" and "@" may be used to edit input.

*** Standalone Format \$Revision: 9.5 \$ ***

Device to format? hd(2,2) (error messages may occur as old bad-sector table is read) Formatting drive hd2: verify (yes/no)? yes Available test patterns are:

- 1 (f00f) rh750 worst case
- 2 (ec6d) media worst case
- 3 (a5a5) alternating 1's and 0's
- 4 (0) zero disk
- 5 (ffff) Severe burn-in (takes several hours)

Select Pattern (one of the above, other to restart) [5]: 2 Device data: #cylinders = 566, #tracks = 7, #sectors = 36

Existing bad block table is 86 entries long

Change defaults? [n] yes

- 1 Change interleave [4]
- 2 Change first cylinder [0]
- 3 Change last cylinder [565]
- 4 Change number of entries in Bad and Hidden tables[86]
- 5 Print current Bad and Hidden tables
- 6 Add new Bad Block Table entry
- 7 Exit Format (with no more changes to disk)
- 8 Change pattern
- 9 Delete block from Bad and Hidden tables
- 10 Convert block number to cyl, track, sector
- 11 Convert cyl, track, sector to block number
- 12 Number of passes [1]
- 13 Write Bad and Hidden tables every [0] cylinders
- 14 Interactive addition of bad blocks
- 15 Number of times to retry save read [8]
- 16 Change configuration record

Any other value exits change menu and starts FORMAT

Block numbers can be either 'nnn' or 'ccc.ttt.sss' (1234 or 1.0.1)

Option: 0

Attempt to preserve existing data (yes/no)? [y] yes Start FORMAT (yes/no)? yes

FORMAT of hd(2,2) ... [cylinders 0 through 565]

FORMATTING CYLINDER 565

A total of 86 bad blocks were found A total of 0 of these were found on this format FORMAT(8R) FORMAT(8R)

Contents of Bad Block Table:

(the contents are listed)

Writing Bad Block Table at block number 8

Format done ...

Exit called - Press Enter to return to main menu

DIAGNOSTICS

The diagnostics are intended to be self-explanatory.

USING THE DIAGNOSTICS DISKETTE TO FORMAT

You should shut down IBM/4.3 and halt the machine to do any disk formatting. Note that the diagnostics diskette format routine does not attempt to save data.

SEE ALSO

sautil(8R), disk(4), hd(4), newfs(8)

BUGS

It should be possible to specify that the bad-block table be read from some other file or device so that it can easily be restored from a backup copy in case of loss.

Format does not format SCSI disks.

HALT(8)

NAME

halt - stop the processor

SYNOPSIS

$$/etc/halt[-n][-q][-y]$$

DESCRIPTION

Halt writes out sandbagged information to the disks and then stops the processor. The machine does not reboot.

The -n option prevents the sync before stopping. The -q option causes a quick halt, no graceful shutdown is attempted. The -y option is needed if you are trying to halt the system from a dialup.

When the LEDs on the machine go out, it is safe to turn it off.

SEE ALSO

reboot(8), shutdown(8)

IBM3812PP(8) IBM3812PP(8)

NAME

ibm3812pp - IBM 3812 Pageprinter server

SYNOPSIS

```
/usr/lib/p3812/ibm3812pp [ -d ] [ -b baudrate ] [ -s socket ] [ -S status ] [ -D device ]
```

DESCRIPTION

Ibm3812pp is the print server daemon for the IBM 3812 Pageprinter. It is normally started by the spooling filter ppt(8) the first time a job is printed on the IBM 3812. There is one print server daemon per 3812 device. The print server opens communications with the printer using the ap(4) line discipline.

These options are interpreted by ibm38/2pp:

-d Debugging statements are written to the log file. Multiple occurrences of this flag increases the amount of debugging information supplied.

- h haudrate

Set the speed of the serial line to the printer to baudrate. The default is 19.2 kb.

-s socket

Socket is the file name to be used in setting up an AF_UNIX domain socket connection to communicate with pp(8). There is one socket per printer. This socket is usually created in the spool file directory for that printer. The default socket name is created by concatenating the suffix "3812" to the name of the printcap(5) entry. For example, the default for the "pp" printer is |usr|spool|ppd|pp3812.

-S status

The status file reflects the current status of the 3812 printer. The default file is |usr|spool|ppd|status3812. This file contains information such as "out of paper".

$-\mathbf{D}$ device

The special file name for the 3812 printer. The default is /dev/pp.

The print server establishes a socket for communications with the spooling system and the 3812 filters using the file name specified with the -s option. The print server uses the system calls listen(2) and select(2) to receive requests from the spooling system filter, ppt, or to time out and query the printer for status. The printer status is maintained in the file identified by the -S option. If a request is received from the 3812 filter (the client), an accept(2) is issued. It will process one client at a time before listening for the next connect(2). The first data received from the client is a structure containing the type of data being transferred, and a list of printer error conditions for which the client will receive notification. This is in the print_3812_flags data structure (see printer3812(5)). This structure is re-written to the client to verify that communications have been correctly established. All subsequent data from the client are written to the 3812 Page-printer. Any messages received from the 3812 are written on the socket to the client if the client requested that it wanted notification.

Two types of data are accepted: ASCII data and Page Map Primitive (PMP) commands. The print server writes ASCII data directly to the 3812 printer, and writes PMP data after inserting ASCII escapes.

FILES

/dev/pp 3812 printer

/usr/spool/*/*3812 name of AF UNIX domain socket for 3812 printer

/usr/spool/*/status3812 status for 3812 printer

/usr/adm/ppd-errs error log for 3812 print server

SEE ALSO

ap(4), printer3812(5), ppt(8)

IBM 3812 Pageprinter Programming Reference, S544-3268

IFCONFIG(8C) IFCONFIG(8C)

NAME

ifconfig - configure network interface parameters

SYNOPSIS

/etc/ifconfig interface address_family [address [dest_address]] [parameters] /etc/ifconfig interface [protocol family]

DESCRIPTION

Ifconfig is used to assign an address to a network interface and/or configure network interface parameters. Ifconfig must be used at boot time to define the network address of each interface present on a machine; it may also be used at a later time to redefine an interface's address or other operating parameters. The interface parameter is a string of the form "name unit", e.g. "en0".

Since an interface may receive transmissions in differing protocols, each of which may require separate naming schemes, it is necessary to specify the *address family*, which may change the interpretation of the remaining parameters. The address families currently supported are "inet" and "ns".

For the DARPA-Internet family, the address is either a host name present in the host name data base, hosts(5), or a DARPA Internet address expressed in the Internet standard "dot notation". For the Xerox Network Systems(tm) family, addresses are net:a.b.c.d.e.f, where net is the assigned network number (in decimal), and each of the six bytes of the host number, a through f, are specified in hexadecimal. The host number may be omitted on 10Mb/s Ethernet interfaces, which use the hardware physical address, and on interfaces other than the first.

The following parameters may be set with ifconfig:

up

Mark an interface "up". This may be used to enable an interface after an "ifconfig down." It happens automatically when setting the first address on an interface. If the interface was reset when previously marked down, the hardware will be re-initialized.

down

Mark an interface "down". When an interface is marked "down", the system will not attempt to transmit messages through that interface. If possible, the interface will be reset to disable reception as well. This action does not automatically disable routes using the interface.

trailers

Request the use of a "trailer" link level encapsulation when sending (default). If a network interface supports trailers, the system will, when possible, encapsulate outgoing messages in a manner which minimizes the number of memory to memory copy operations performed by the receiver. On networks that support the Address Resolution Protocol (see arp(4P); currently, only 10 Mb/s Ethernet), this flag indicates that the system should request that other systems use trailers when sending to this host. Similarly, trailer encapsulations will be sent to other hosts that have made such requests. Currently used by Internet protocols only.

- trailers

Disable the use of a "trailer" link level encapsulation.

arp

Enable the use of the Address Resolution Protocol in mapping between network level addresses and link level addresses (default). This is currently implemented for mapping between DARPA Internet addresses and 10Mb/s Ethernet addresses.

– arp

Disable the use of the Address Resolution Protocol.

metric n

Set the routing metric of the interface to n, default 0. The routing metric is used by the routing protocol (routed(8c)). Higher metrics have the effect of making a route less favorable; metrics are counted as additional hops to the destination network or host.

IFCONFIG(8C) IFCONFIG(8C)

debug Enable driver dependent debugging code; usually, this turns on extra console

error logging.

- debug Disable driver dependent debugging code.

netmask mask (Inet only) Specify how much of the address to reserve for subdividing networks

into sub-networks. The mask includes the network part of the local address and the subnet part, which is taken from the host field of the address. The mask can be specified as a single hexadecimal number with a leading 0x, with a dotnotation Internet address, or with a pseudo-network name listed in the network table networks(5). The mask contains 1's for the bit positions in the 32-bit address which are to be used for the network and subnet parts, and 0's for the host part. The mask should contain at least the standard network portion, and

the subnet field should be contiguous with the network portion.

dstaddr Specify the address of the correspondent on the other end of a point to point

link.

broadcast (Inet only) Specify the address to use to represent broadcasts to the network.

The default broadcast address is the address with a host part of all 1's.

ipdst (NS only) This is used to specify an Internet host who is willing to receive ip

packets encapsulating NS packets bound for a remote network. In this case, an apparent point to point link is constructed, and the address specified will be

taken as the NS address and network of the destinee.

bridge Enable routing field support. This is necessary to communicate across a bridge,

or with any workstation supporting token ring routing fields, such as AIX. This

is now the configuration default.

-bridge Disable routing field support. This is included for compatibility with earlier 4.2

versions, none of which support routing fields on the ring.

snap Send token ring headers with the extended snap format. This is the current

IEEE standard, and necessary to communicate with any machine using this for-

mat, such as AIX. This is now the configuration default.

-snap Send token ring headers in the old format. This is included for compatibility

with earlier 4.2 versions.

If config displays the current configuration for a network interface when no optional parameters are supplied. If a protocol family is specified, if config will report only the details specific to that protocol family.

Only the super-user may modify the configuration of a network interface.

DIAGNOSTICS

Messages indicating the specified interface does not exit, the requested address is unknown, or the user is not privileged and tried to alter an interface's configuration.

SEE ALSO

netstat(1), intro(4N), rc(8)

NAME

init - process control initialization

SYNOPSIS

/etc/init

DESCRIPTION

Init is invoked inside IBM/4.3 as the last step in the boot procedure. It then normally runs the automatic reboot sequence as described in reboot(8) and, if this succeeds, begins multi-user operation. If the reboot fails, init commences single-user operation by giving the super-user a shell on the console. It is possible to pass parameters from the boot program to init so that single-user operation commences immediately. When single-user operation is terminated by killing the single-user shell (i.e. by hitting ^D), init runs /etc/rc without the reboot parameter. This command file performs housekeeping operations such as removing temporary files, mounting file systems and starting daemons.

In multi-user operation, *init*'s role is to create a process for each terminal port on which a user may log in. To begin such operations, it reads the file /etc/ttys and executes a command for each terminal specified in the file. This command will usually be /etc/getty. Getty opens and initializes the terminal line, reads the user's name, and invokes login to log in the user and execute the shell.

Ultimately, the shell will terminate because of an end-of-file that is either typed explicitly or generated as a result of hanging up. The main path of *init*, which has been waiting for such an event, removes the appropriate entry from the file *utmp*, which records current users, and makes an entry in /usr/adm/wtmp, which maintains a history of logins and logouts. The *wtmp* entry is made only if a user logged in successfully on the line. The appropriate terminal is then reopened and getty(8) is reinvoked.

Init catches the hangup signal (signal SIGHUP) and interprets it to mean that the file /etc/ttys should be read again. The shell process on each line that was active but no longer exists in ttys is terminated; a new process is created for each added line; lines unchanged in the file are undisturbed. Thus it is possible to drop or add phone lines without rebooting the system by changing the ttys file and sending a hangup signal to the init process: use "kill - HUP 1".

Init will terminate multi-user operation and resume single-user mode if sent a terminate (TERM) signal, i.e. "kill -TERM 1". If there are processes outstanding that are deadlocked (because of hardware or software failure), init will not wait for them all to die (which might take forever), but will time out after 30 seconds and print a warning message.

Init will cease creating new getty(8)s and allows the system to die slowly if it receives a terminal stop (TSTP) signal, i.e. "kill -TSTP 1". A later hangup will resume full multi-user operation, or a terminate will start a single-user shell. This hook is used by reboot(8) and halt(8).

Init's role is critical; if it dies, the system reboots itself automatically. If, at bootstrap time, the init process cannot be located, the system prints the message "can't exec /etc/init", and hangs.

DIAGNOSTICS

/etc/getty gettyargs failing, sleeping. A process being started to service a line is exiting quickly each time it is started. This is often caused by a ringing or noisy terminal line. *Init* will sleep for 30 seconds.

WARNING: Something is hung (won't die); ps axl advised. A process hung and could not be killed when the system was shutting down. This is usually caused by a process stuck in a device driver because of a persistent device-error condition.

FILES

/dev/console /dev/tty* /etc/utmp /usr/adm/wtmp INIT(8)

/etc/ttys /etc/rc

SEE ALSO

login(1), kill(1), sh(1), ttys(5), crash(8R), getty(8), rc(8), reboot(8), halt(8), shutdown(8)

LANDUMP(8R) LANDUMP(8R)

NAME

landump - dump IBM Token-Ring Personal Computer Adapter

SYNOPSIS

/etc/landump[-d]interface

DESCRIPTION

Landump is used to freeze the IBM token ring adapter and dump the contents of the adapter's internal storage for diagnostic purposes.

The interface parameter is a string of the form "name unit", for example, "lan0". If no options are specified, the adapter is frozen and dumped. If the -d option is specified, landump runs as a daemon, waiting until a dump is required due to an event internal to the adapter, for instance, adapter hardware failure. When such an event occurs, landump is notified by the adapter driver and the adapter is frozen and the dump occurs at that time. The daemon then waits for the next event

The contents of the dump are written to a file named interfacecore, for example, "lan0core", in the current directory.

DIAGNOSTICS

Messages indicating the specified interface does not exist, or the user is not privileged.

SEE ALSO

lan(4), rc(8)

BUGS

When run as a daemon, *landump* doesn't place itself in the background. It must be explicitly placed in the background.

LPFILTER(8R)

LPFILTER(8R)

NAME

ibmbit, ibmgra, ibmpro — output filters for the IBM 4201 Proprinter and IBM 5152 Graphics Printer

SYNOPSIS

/usr/lib/ibmlp/ibmbit [-f] /usr/lib/ibmlp/ibmgra /usr/lib/ibmlp/ibmpro

DESCRIPTION

These functions are not normally called by the user, but invoked as output filters by lpr(1).

Ibmbit is invoked as the -v (raster data) output filter by lpr(1) when called by bitprt(1). Ibmbit reads a bitmap image file (see dumpapa16(1)) from the standard input, rotates it by 90 degrees, adds appropriate escape control sequences, and writes it to the standard output for printing on the local line printer (either an IBM 4201 Proprinter or IBM 5152 Graphics Printer). 90-degree rotation is performed to let the aspect ratio of the printed page mirror that of the screen. Ibmbit defaults to providing the highest print quality available by printing at half-speed. The user can use normal speed, at a sacrifice of print quality, by invoking ibmbit - f manually, and directing the output to |dev|lp?.

Ibmgra is invoked as the -t (troff data) output filter by lpr(1) when called by proff(1). It is used to pass nroff(1) output to the IBM 5152 Graphics Printer.

Ibmpro is invoked as the $-\mathbf{n}$ (nroff data) output filter by lpr(1) when called by proff(1). It is used to pass nroff(1) output to the IBM 4201 Proprinter.

SEE ALSO

bitprt(1), dumpaed(1), dumpaea16(1), dumpaea8(1), dumpaea8c(1), lpr(1), scale(1), lp(4) R. Campbell, "4.3BSD Line Printer Spooler Manual", in UNIX System Manager's Manual

BUGS

Rotation of the image by *ibmbit* is not optional.

MAKEDEV(8) MAKEDEV(8)

NAME

makedev - make system special files

SYNOPSIS

/dev/MAKEDEV device-name? ...

DESCRIPTION

MAKEDEV is a shell script normally used to install special files. It resides in the /dev directory, the normal location of special files. Arguments to MAKEDEV are usually of the form device-name?, where device-name is one of the supported devices listed in section 4 of the manual and "?" is a logical unit number (0-9). A few special arguments create assorted collections of devices and are listed below.

std Create the standard devices for the system; e.g. |dev|console, |dev|tty.

local Create those devices specific to the local site. This request causes the shell file /dev/MAKEDEV.local to be executed. Site-specific commands, such as those used to setup dialup lines as ttyd?, should be included in this file.

displays

Create all the console display devices.

conttys Create a tty device for each console display.

mice Create all planar mouse/tablet input devices.

X Create the devices needed by the X window manager; e.g. contrys, mice, and displays.

floorstand

Create devices for standard configuration for the floorstand configuration.

Since all devices are created using mknod(8), this shell script is useful only to the super-user.

DIAGNOSTICS

Diagnostics are either self-explanatory or generated by one of the programs called from the script. Use "sh -x MAKEDEV" in case of trouble.

SEE ALSO

intro(4), config(8), mknod(8)

BUGS

When more than one piece of hardware of the same "kind" is present on a machine, naming conflicts arise.

MAKESYM(8) MAKESYM(8)

NAME

makesym - make debugger symbol table

SYNOPSIS

```
makesym [-n][-o \text{ output }][-h][\text{ input }]
```

DESCRIPTION

Makesym converts the output of nm(1) into the format accepted by the standalone debugger so that it may be include with the program being debugged. This is most often used to provide a symbol table for debugging the kernel.

The output is the symbol table, either in raw form (useful when downloading over a serial line) or as an a.out file that can be merged into a program.

The following switches are accepted:

- -o the following argument specifies an output file. If this switch is not specified, the result will appear on standard output.
- -h produces an a.out header. It must appear after -o (if that switch is present).
- -n does *not* byte-swap the address produced (only effective when run on a VAX or other machine that requires byte-swapping).

Makesym is written to accept the output of nm rather than reading the symbol table directly from an a.out format file. This allows easy deletion and editing of the symbol table using normal tools (sed, grep,etc.).

SEE ALSO

nm(1), config(8), debug(8)

IBM3812PP(8) IBM3812PP(8)

NAME

ibm3812pp - IBM 3812 Pageprinter server

SYNOPSIS

```
/usr/lib/p3812/ibm3812pp [-d][-b baudrate ][-s socket ][-S status ][-D device ][-S status ][-D device ][-D
```

DESCRIPTION

Ibm3812pp is the print server daemon for the IBM 3812 Pageprinter. It is normally started by the spooling filter ppt(8) the first time a job is printed on the IBM 3812. There is one print server daemon per 3812 device. The print server opens communications with the printer using the ap(4) line discipline.

These options are interpreted by ibm3812pp:

-d Debugging statements are written to the log file. Multiple occurrences of this flag increases the amount of debugging information supplied.

-b baudrate

Set the speed of the serial line to the printer to baudrate. The default is 19.2 kb.

-s socket

Socket is the file name to be used in setting up an AF_UNIX domain socket connection to communicate with pp(8). There is one socket per printer. This socket is usually created in the spool file directory for that printer. The default socket name is created by concatenating the suffix "3812" to the name of the printcap(5) entry. For example, the default for the "pp" printer is |usr|spool|ppd|pp3812.

-S status

The status file reflects the current status of the 3812 printer. The default file is |usr|spool|ppd|status3812. This file contains information such as "out of paper".

−**D** device

The special file name for the 3812 printer. The default is /dev/pp.

The print server establishes a socket for communications with the spooling system and the 3812 filters using the file name specified with the -s option. The print server uses the system calls listen(2) and select(2) to receive requests from the spooling system filter, ppt, or to time out and query the printer for status. The printer status is maintained in the file identified by the -S option. If a request is received from the 3812 filter (the client), an accept(2) is issued. It will process one client at a time before listening for the next connect(2). The first data received from the client is a structure containing the type of data being transferred, and a list of printer error conditions for which the client will receive notification. This is in the print_3812 flags data structure (see printer3812(5)). This structure is re-written to the client to verify that communications have been correctly established. All subsequent data from the client are written to the 3812 Page-printer. Any messages received from the 3812 are written on the socket to the client if the client requested that it wanted notification.

Two types of data are accepted: ASCII data and Page Map Primitive (PMP) commands. The print server writes ASCII data directly to the 3812 printer, and writes PMP data after inserting ASCII escapes.

FILES

/dev/pp 3812 printer
/usr/spool/*/*3812 name of AF_UNIX domain socket for 3812 printer
/usr/spool/*/status3812 status for 3812 printer
/usr/adm/ppd-errs error log for 3812 print server

SEE ALSO

ap(4), printer3812(5), ppt(8)

IBM 3812 Pageprinter Programming Reference, S544-3268

IFCONFIG(8C) IFCONFIG(8C)

NAME

ifconfig - configure network interface parameters

SYNOPSIS

/etc/ifconfig interface address_family [address [dest_address]] [parameters]
/etc/ifconfig interface [protocol family]

DESCRIPTION

Ifconfig is used to assign an address to a network interface and/or configure network interface parameters. Ifconfig must be used at boot time to define the network address of each interface present on a machine; it may also be used at a later time to redefine an interface's address or other operating parameters. The interface parameter is a string of the form "name unit", e.g. "en0".

Since an interface may receive transmissions in differing protocols, each of which may require separate naming schemes, it is necessary to specify the *address_family*, which may change the interpretation of the remaining parameters. The address families currently supported are "inet" and "ns".

For the DARPA-Internet family, the address is either a host name present in the host name data base, hosts(5), or a DARPA Internet address expressed in the Internet standard "dot notation". For the Xerox Network Systems(tm) family, addresses are net:a.b.c.d.e.f, where net is the assigned network number (in decimal), and each of the six bytes of the host number, a through f, are specified in hexadecimal. The host number may be omitted on 10Mb/s Ethernet interfaces, which use the hardware physical address, and on interfaces other than the first.

The following parameters may be set with *ifconfig*:

up

Mark an interface "up". This may be used to enable an interface after an "ifconfig down." It happens automatically when setting the first address on an interface. If the interface was reset when previously marked down, the hardware will be re-initialized.

down

Mark an interface "down". When an interface is marked "down", the system will not attempt to transmit messages through that interface. If possible, the interface will be reset to disable reception as well. This action does not automatically disable routes using the interface.

trailers

Request the use of a "trailer" link level encapsulation when sending (default). If a network interface supports trailers, the system will, when possible, encapsulate outgoing messages in a manner which minimizes the number of memory to memory copy operations performed by the receiver. On networks that support the Address Resolution Protocol (see arp(4P); currently, only 10 Mb/s Ethernet), this flag indicates that the system should request that other systems use trailers when sending to this host. Similarly, trailer encapsulations will be sent to other hosts that have made such requests. Currently used by Internet protocols only.

- trailers

Disable the use of a "trailer" link level encapsulation.

arp

Enable the use of the Address Resolution Protocol in mapping between network level addresses and link level addresses (default). This is currently implemented for mapping between DARPA Internet addresses and 10Mb/s Ethernet addresses.

– arp

Disable the use of the Address Resolution Protocol.

metric n

Set the routing metric of the interface to n, default 0. The routing metric is used by the routing protocol (routed(8c)). Higher metrics have the effect of making a route less favorable; metrics are counted as additional hops to the destination network or host.

IFCONFIG(8C) IFCONFIG(8C)

debug Enable driver dependent debugging code; usually, this turns on extra console

error logging.

- debug Disable driver dependent debugging code.

netmask mask (Inet only) Specify how much of the address to reserve for subdividing networks

into sub-networks. The mask includes the network part of the local address and the subnet part, which is taken from the host field of the address. The mask can be specified as a single hexadecimal number with a leading 0x, with a dot-notation Internet address, or with a pseudo-network name listed in the network table networks(5). The mask contains 1's for the bit positions in the 32-bit address which are to be used for the network and subnet parts, and 0's for the host part. The mask should contain at least the standard network portion, and

the subnet field should be contiguous with the network portion.

dstaddr Specify the address of the correspondent on the other end of a point to point

link.

broadcast (Inet only) Specify the address to use to represent broadcasts to the network.

The default broadcast address is the address with a host part of all 1's.

ipdst (NS only) This is used to specify an Internet host who is willing to receive ip

packets encapsulating NS packets bound for a remote network. In this case, an apparent point to point link is constructed, and the address specified will be

taken as the NS address and network of the destinee.

bridge Enable routing field support. This is necessary to communicate across a bridge,

or with any workstation supporting token ring routing fields, such as AIX. This

is now the configuration default.

-bridge Disable routing field support. This is included for compatibility with earlier 4.2

versions, none of which support routing fields on the ring.

snap Send token ring headers with the extended snap format. This is the current

IEEE standard, and necessary to communicate with any machine using this for-

mat, such as AIX. This is now the configuration default.

-snap Send token ring headers in the old format. This is included for compatibility

with earlier 4.2 versions.

If config displays the current configuration for a network interface when no optional parameters are supplied. If a protocol family is specified, if config will report only the details specific to that protocol family.

Only the super-user may modify the configuration of a network interface.

DIAGNOSTICS

Messages indicating the specified interface does not exit, the requested address is unknown, or the user is not privileged and tried to alter an interface's configuration.

SEE ALSO

netstat(1), intro(4N), rc(8)

INIT(8)

NAME

init - process control initialization

SYNOPSIS

/etc/init

DESCRIPTION

Init is invoked inside IBM/4.3 as the last step in the boot procedure. It then normally runs the automatic reboot sequence as described in reboot(8) and, if this succeeds, begins multi-user operation. If the reboot fails, init commences single-user operation by giving the super-user a shell on the console. It is possible to pass parameters from the boot program to init so that single-user operation commences immediately. When single-user operation is terminated by killing the single-user shell (i.e. by hitting ^D), init runs /etc/rc without the reboot parameter. This command file performs housekeeping operations such as removing temporary files, mounting file systems and starting daemons.

In multi-user operation, init's role is to create a process for each terminal port on which a user may log in. To begin such operations, it reads the file /etc/ttys and executes a command for each terminal specified in the file. This command will usually be /etc/getty. Getty opens and initializes the terminal line, reads the user's name, and invokes login to log in the user and execute the shell.

Ultimately, the shell will terminate because of an end-of-file that is either typed explicitly or generated as a result of hanging up. The main path of *init*, which has been waiting for such an event, removes the appropriate entry from the file *utmp*, which records current users, and makes an entry in /usr/adm/wtmp, which maintains a history of logins and logouts. The *wtmp* entry is made only if a user logged in successfully on the line. The appropriate terminal is then reopened and *getty*(8) is reinvoked.

Init catches the hangup signal (signal SIGHUP) and interprets it to mean that the file /etc/ttys should be read again. The shell process on each line that was active but no longer exists in ttys is terminated; a new process is created for each added line; lines unchanged in the file are undisturbed. Thus it is possible to drop or add phone lines without rebooting the system by changing the ttys file and sending a hangup signal to the init process: use "kill - HUP 1".

Init will terminate multi-user operation and resume single-user mode if sent a terminate (TERM) signal, i.e. "kill -TERM 1". If there are processes outstanding that are deadlocked (because of hardware or software failure), init will not wait for them all to die (which might take forever), but will time out after 30 seconds and print a warning message.

Init will cease creating new getty(8)s and allows the system to die slowly if it receives a terminal stop (TSTP) signal, i.e. "kill -TSTP 1". A later hangup will resume full multi-user operation, or a terminate will start a single-user shell. This hook is used by reboot(8) and halt(8).

Init's role is critical; if it dies, the system reboots itself automatically. If, at bootstrap time, the init process cannot be located, the system prints the message "can't exec /etc/init", and hangs.

DIAGNOSTICS

/etc/getty gettyargs failing, sleeping. A process being started to service a line is exiting quickly each time it is started. This is often caused by a ringing or noisy terminal line. *Init* will sleep for 30 seconds.

WARNING: Something is hung (won't die); ps axl advised. A process hung and could not be killed when the system was shutting down. This is usually caused by a process stuck in a device driver because of a persistent device-error condition.

FILES

/dev/console /dev/tty* /etc/utmp /usr/adm/wtmp

```
INIT(8)
```

INIT(8)

/etc/ttys /etc/rc

SEE ALSO

login(1), kill(1), sh(1), ttys(5), crash(8R), getty(8), rc(8), reboot(8), halt(8), shutdown(8)

LANDUMP(8R) LANDUMP(8R)

NAME

landump - dump IBM Token-Ring Personal Computer Adapter

SYNOPSIS

/etc/landump [-d] interface

DESCRIPTION

Landump is used to freeze the IBM token ring adapter and dump the contents of the adapter's internal storage for diagnostic purposes.

The interface parameter is a string of the form "name unit", for example, "lan0". If no options are specified, the adapter is frozen and dumped. If the -d option is specified, landump runs as a daemon, waiting until a dump is required due to an event internal to the adapter, for instance, adapter hardware failure. When such an event occurs, landump is notified by the adapter driver and the adapter is frozen and the dump occurs at that time. The daemon then waits for the next event.

The contents of the dump are written to a file named interfacecore, for example, "lan0core", in the current directory.

DIAGNOSTICS

Messages indicating the specified interface does not exist, or the user is not privileged.

SEE ALSO

lan(4), rc(8)

BUGS

When run as a daemon, landump doesn't place itself in the background. It must be explicitly placed in the background.

LPFILTER(8R) LPFILTER(8R)

NAME

ibmbit, ibmgra, ibmpro — output filters for the IBM 4201 Proprinter and IBM 5152 Graphics Printer

SYNOPSIS

/usr/lib/ibmlp/ibmbit [-f]
/usr/lib/ibmlp/ibmgra
/usr/lib/ibmlp/ibmpro

DESCRIPTION

These functions are not normally called by the user, but invoked as output filters by lpr(1).

Ibmbit is invoked as the $-\mathbf{v}$ (raster data) output filter by lpr(1) when called by bitprt(1). Ibmbit reads a bitmap image file (see dumpapa16(1)) from the standard input, rotates it by 90 degrees, adds appropriate escape control sequences, and writes it to the standard output for printing on the local line printer (either an IBM 4201 Proprinter or IBM 5152 Graphics Printer). 90-degree rotation is performed to let the aspect ratio of the printed page mirror that of the screen. Ibmbit defaults to providing the highest print quality available by printing at half-speed. The user can use normal speed, at a sacrifice of print quality, by invoking $ibmbit - \mathbf{f}$ manually, and directing the output to $|dev|/lp^2$.

Ibmgra is invoked as the $-\mathbf{t}$ (troff data) output filter by lpr(1) when called by proff(1). It is used to pass nroff(1) output to the IBM 5152 Graphics Printer.

Ibmpro is invoked as the $-\mathbf{n}$ (nroff data) output filter by lpr(1) when called by proff(1). It is used to pass nroff(1) output to the IBM 4201 Proprinter.

SEE ALSO

bitprt(1), dumpaed(1), dumpapa16(1), dumpapa8(1), dumpapa8c(1), lpr(1), scale(1), lp(4) R. Campbell, "4.3BSD Line Printer Spooler Manual", in UNIX System Manager's Manual

BUGS

Rotation of the image by *ibmbit* is not optional.

MAKEDEV(8) MAKEDEV(8)

NAME

makedev - make system special files

SYNOPSIS

/dev/MAKEDEV device-name? ...

DESCRIPTION

MAKEDEV is a shell script normally used to install special files. It resides in the /dev directory, the normal location of special files. Arguments to MAKEDEV are usually of the form device-name?, where device-name is one of the supported devices listed in section 4 of the manual and "?" is a logical unit number (0-9). A few special arguments create assorted collections of devices and are listed below.

std Create the standard devices for the system; e.g. /dev/console, /dev/tty.

local Create those devices specific to the local site. This request causes the shell file /dev/MAKEDEV.local to be executed. Site-specific commands, such as those used to setup dialup lines as ttyd?, should be included in this file.

displays

Create all the console display devices.

conttys Create a tty device for each console display.

mice Create all planar mouse/tablet input devices.

X Create the devices needed by the X window manager; e.g. contrys, mice, and displays.

floorstand

Create devices for standard configuration for the floorstand configuration.

Since all devices are created using mknod(8), this shell script is useful only to the super-user.

DIAGNOSTICS

Diagnostics are either self-explanatory or generated by one of the programs called from the script. Use "sh -x MAKEDEV" in case of trouble.

SEE ALSO

intro(4), config(8), mknod(8)

BUGS

When more than one piece of hardware of the same "kind" is present on a machine, naming conflicts arise.

MAKESYM(8) MAKESYM(8)

NAME

makesym - make debugger symbol table

SYNOPSIS

```
makesym [-n][-o \text{ output }][-h][\text{ input }]
```

DESCRIPTION

Makesym converts the output of nm(1) into the format accepted by the standalone debugger so that it may be include with the program being debugged. This is most often used to provide a symbol table for debugging the kernel.

The output is the symbol table, either in raw form (useful when downloading over a serial line) or as an a.out file that can be merged into a program.

The following switches are accepted:

- the following argument specifies an output file. If this switch is not specified, the result will appear on standard output.
- -h produces an a.out header. It must appear after -o (if that switch is present).
- -n does *not* byte-swap the address produced (only effective when run on a VAX or other machine that requires byte-swapping).

Makesym is written to accept the output of nm rather than reading the symbol table directly from an a.out format file. This allows easy deletion and editing of the symbol table using normal tools (sed, grep,etc.).

SEE ALSO

nm(1), config(8), debug(8)

NAME

minidisk - minidisk maintenance utility

DESCRIPTION

Minidisk is a component of the sautil(8R) package of standalone utility programs and is used in maintaining VRM (Virtual Resource Manager) compatible minidisk (partition) tables. Minidisk displays and allows changes to be made to the minidisk directory. It can be used to change the size and location of the IBM 4.3 hard disk partitions. On the IBM 6152 Academic System, mindisk should be used after an IBM/4.3 physical partition has been created with the standalone utility program fdisk(8R).

When AIX (Advanced Interactive Executive) is to be used on the same machine with IBM/4.3, the IBM/4.3 disks must have minidisk tables to insure on the IBM RT PC VRM and AIX do not overwrite the IBM/4.3 disks.

When first invoked *minidisk* will prompt for the name of the disk to be examined or changed. See the section below under the *disk* command for format of the response.

Minidisk Commands

There are a number of minidisk commands available:

change name field value

changes the value of the field (one of name, iodn, type) specified. The value must be appropriate for the field specified.

create name iodn size type

creates a minidisk with the appropriate name (see NOTE below for the acceptable names). Iodn should be in the range 32736-32751 which is allocated to IBM/4.3. Size is the number of (512-byte) blocks to allocate to the partition. Type is the type of the partition; IBM/4.3 partitions have a type of zero, except for the swap/paging area which has type swap. Type can be specified in hexadecimal (see IBM RT PC Virtual Resource Manager Virtual Resource Manager Technical Reference), or as a comma-separated list of types from ipl, vrm, pc, aix, file, swap and nobad.

delete name

deletes minidisk name, after prompting the user for confirmation.

directory

prints the current directory.

disk disk-name

specifies a new disk to be examined. When used in standalone mode, the *disk-name* should be of the form: hd(unit,2), where unit is one of 0, 1, or 2. When used as a normal utility (this is generally not recommended as the kernel disk driver will not pick up any partition changes until the system is rebooted), disk-name is of the form: /dev/rhdunitc for the hard disk, /dev/rscunitc for the IBM 9331 disk unit (scsi's).

help

prints out a short description of each command.

initialize

creates an empty minidisk table (for example, makes all the space on the disk available for use). You are prompted for confirmation as this command will make any partitions previously set up unavailable.

list

prints (lists) the current directory. This is a synonym for the directory command.

quit

quits minidisk and returns control to sautil(8R).

standard

creates the standard (default) IBM/4.3 partitions (boot, a, b, and g). For this command to be accepted, an *initialize* command must be given previously.

swap

swaps the primary and secondary copies of the directory. This command is only effective if the copies were initially different (which is usually because either the primary or secondary copy was damaged). By using this command, it is possible to examine both copies to determine which is best used. After selecting the desired copy, the *write* command can be used to force both copies to be written to the disk. This can also be accomplished by making a change to any minidisk entry.

write

forces the current minidisk record to be written to the disk. As this is done after any change to the directory, this command is useful with the *swap* command after either the primary or secondary directory copy was damaged.

NOTE

On the IBM RT PC, an IBM/4.3 minidisk must have a name of the form "hdxy" for the hard disk, "scxy" for the scsi to be taken as a IBM/4.3 disk partition, where x is any character, and y is a letter between "a" and "g". On the IBM 6152 Academic System, and IBM/4.3 mindisk must have a name of the form "hdxy" to be taken as an IBM/4.3 disk partition, where x is any character, and y is a letter between "a" and "g". Note that the drivers ignore a "c" partition. The "c" partition always refers to the whole disk. In addition, any minidisk with the swap type is used as a IBM/4.3 "b" partition (for swapping/paging).

Although minidisks can be created with any size at any location on the disk, the IBM/4.3 disk drivers adjust the start address to be on the next full cylinder boundary and adjust the size accordingly.

When *minidisk* is used on a disk for the first time, it is likely a "minidisk corrupted" message will be printed; answer "yes" when asked if the minidisk should be initialized, if this is the case.

The defaults used for root and swap are 15884 and 10032 or about 7.5M and 5M. Users are left to decide their disk space requirements and make adjustments if necessary.

IODNs and minidisk types have little meaning in the System/2 environment. An ipl minidisk will (harmlessly) not be created on a drive with less than 68 sectors per cylinder.

EXAMPLE

In this example, minidisk is loaded from the sautil menu. Boldface indicates user input.

4.3 BSD UNIX Standalone Maintenance Program \$Revision: 9.7 \$

Choice Description

- 1 boot boot standalone program or kernel
- 2 format format hard disk
- 3 dump display disk or diskette (hex)
- 4 cat display a file contents (ASCII)
- 5 ls print directory of filesystem
- 6 copy copy all/part of disk or diskette
- 7 debugger display memory, etc.
- 8 iplsource set boot order in nyram
- 9 minidisk display/change minidisk directory
- 10 dosboot boot standalone program or kernel from DOS diskette

```
Enter the menu choice number desired, then press < Enter >
        Choice: 9
        disk
                hd(0,2)
        hd(0,2):
        number = 0 level = 0 first = 0 last = 0 bad_block = 0 bad_size = 0
       index iodn name
                                              start size type
                                date
        minidisk directory corrupted
        re-initialize minidisk directory [y/n] y
        > list
        number = 1 level = 0 first = 0 last = 0 bad block = 138127 bad size = 1000
       index iodn name
                                date
                                              start size type
           0
               0 FREE
                                              68 138060 00
        > standard
        create boot 32736 68 ipl
        create hd0a 32737 15912 0
        create hd0b 32738 33456 swap
       create hd0g 32739 88536 0
        > list
       number = 5 level = 0 first = 0 last = 4 bad block = 138127 bad size = 1000
       index iodn name
                                              start size type
                                date
           0 32736 boot Thu Jul 31 17:45:09 1986
                                                           68 01 ipl
                                                     68
           1 32737 hd0a Thu Jul 31 17:45:09 1986
                                                    136 15912 00
           2 32738 hd0b Thu Jul 31 17:45:09 1986 16048 33456 20 swap
           3 32739 hd0g Thu Jul 31 17:45:09 1986 49504 88536 00
           4
               0 FREE
                                           138040
                                                     88 00
        > help
        Commands available are:
       change
                        change minidisk parameters
       create
                       create new minidisk
       delete
                       delete minidisk directory
       dir
                       list minidisk directory
       disk
                       look at different disk
                       display short help messages about each command
       help
       initialize
                       initialize (erase) entire minidisk directory
                       list minidisk directory
       list
                       leave minidisk program
       quit
                       create standard partitions
       standard
                       use secondary minidisk (if primary was bad)
       swap
                       force minidisk directory to disk
       write
        > quit
Minidisk cannot be used from sautil to change the disk partitions on an scsi.
boot(8), debug(8), format(8R), sautil(8R)
```

BUGS

SEE ALSO

IBM RT PC Virtual Resource Manager Virtual Resource Manager Technical Reference, SV21-

8013

IBM RT PC Advanced Interactive Executive Operating System AIX Operating System Technical Reference, SV21-8009

Section 5, "AIX and IBM/4.3 Co-residence" in "Installing and Operating Academic Information System 4.3" in Volume II, Supplementary Documents

NEWFS(8) NEWFS(8)

NAME

newfs - construct a new file system

SYNOPSIS

/etc/newfs [-v][-n][-N][mkfs-options] special [disk-type]

DESCRIPTION

Newfs is a "friendly" front-end to the mkfs(8) program. Newfs looks up the type of disk a file system is being created on in the disk description file |etc| disktab and calculates the appropriate parameters to use in calling mkfs. Newfs then builds the file system by forking mkfs, and if the file system is a root partition, installs the necessary bootstrap programs. The -n option prevents the bootstrap programs from being installed.

If the -v option is supplied, newfs will print out its actions, including the parameters passed to mkfs.

The -N option prevents mkfs(8) from being invoked.

If the special file supports the DKIOCGPART ioctl, then the disk-type is optional and the default size is taken from the actual partition size rather than /etc/disktab.

Options which may be used to override default parameters passed to mkfs are:

- -s size The size of the file system in sectors.
- -b block-size

The block size of the file system in bytes.

- f frag-size

The fragment size of the file system in bytes.

- t #tracks/cylinder
- c #cylinders/group

The number of cylinders per cylinder group in a file system. The default value used is 16.

-m free space %

The percentage of space reserved from normal users; the minimum free space threshold. The default value used is 10%.

- r revolutions/minute

The speed of the disk in revolutions per minute (usually 1800 or 3600).

-S sector-size

The size of a sector in bytes (almost never anything but 512).

-i number of bytes per inode

This specifies the density of inodes in the file system. The default is to create an inode for each 2048 bytes of data space. If fewer inodes are desired, a larger number should be used; to create more inodes a smaller number should be given.

FILES

/etc/disktab for disk geometry and file system partition information

/etc/mkfs actually builds the file system

/usr/mdec for boot-strapping programs

SEE ALSO

disk(4), fd(4), hd(4), scsi(4), disktab(5), fs(5), diskpart(8), format(8R), fsck(8), mkfs(8), tunefs(8)

"A Fast File System for Unix" in UNIX System Manager's Manual

NEWVD(8)

NAME

newvd - create a new filesystem on a Remote Virtual Disk (RVD)

SYNOPSIS

newvd drive disk-type

DESCRIPTION

Newvd invokes mkfs(8) and executes it with the appropriate parameters for the physical device on which the RVD resides. Drive is an 1-digit integer indicating the virtual drive on which the RVD is spun up (see vdstats(8)). Disk-type is the type of physical disk, as found in /etc/disktab (see disktab(5)), on which the RVD actually resides.

DIAGNOSTICS

"Virtual drive is unused or off" means the RVD is not spun up. Other messages are self-explanatory.

FILES

/etc/disktab

parameters for disk layouts

SEE ALSO

disktab(5), mkfs(8), newfs(8), vdstats(8)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

OMERGE(8) OMERGE(8)

NAME

omerge - merge object files

SYNOPSIS

omerge [-p#][-D][-l#][-#] input output

DESCRIPTION

Omerge takes two a.out files -- an input file and an output file -- and merges the input file into the output file. This merge makes sense only if space for the input file has been allocated in the output file.

Omerge is generally used to combine two a.out files that cannot be linked by ld in the normal fashion, typically because they are linked to different base addresses, or when one module depends upon the content of the other (such as a built-in symbol table). The current uses for omerge are:

- to combine debugger symbol tables with the module (the symbol table cannot be produced until after the link)
- to combine a debugger (linked for physical addresses) and a kernel (linked for virtual addresses)

The following switches are accepted:

- -p# sets the logical page size, which is required to process ZMAGIC (demand-paged) files properly when cross linking.
- -D prints internal debugging information about what omerge is doing.
- -l# specifies a length that the input file cannot exceed.
- -# specifies a hex offset at which the input file is placed in the output file. If no offset is specified, "0" is assumed.

DIAGNOSTICS

switch %s not recognized

usage: %s infile outfile cannot open %s for input

cannot open %s for input/output

%s: header too short

%s: bad header type

%s: header too short

%s: bad header type

input file exceeds available space by %d bytes

%s: read error

%s: file too short

%s: write error

PPT(8) PPT(8)

NAME

ppt - spooling system filter for the IBM 3812 Pageprinter

SYNOPSIS

/usr/lib/p3812/ppt [- t type] [-w width] [- l length] [- i indent] [- n login] [- h host] [accounting file |

DESCRIPTION

Ppt is the spooling system filter used by the print server daemon (ibm38/2pp(8)) for printing on the IBM 3812 Pageprinter. This filter is identified to the spooling system as the of capability in the printcap(5) entry for the 3812. Ppt is also invoked by the if capability pplpf, and is linked from pmp which is the vf filter. The options and arguments on the ppt command correspond to those defined by the spooling system for filters (see the 4.3BSD Line Printer Spooler Manual). There is one additional option, -t, which identifies the type of data to be processed. Type can be either i for text data or v for Page Map Primitive (PMP) data.

Ppt will first determine the name of the socket to be used to communicate with the print server. The name of this file is constructed from the spool directory name and the printer name. For example, if the spool directory is /usr/spool/ppd and the printer is pp, then the name of the socket is stored in the file /usr/spool/ppd/pp3812. Ppt then attempts to connect to that socket. If the print server, ibm38/2pp, for that socket has not been started, then ppt starts the print server using an execl. It passes the device name to the print server. The device name is the PP capability in the printcap entry for that printer.

Once the connection is made, ppt sends the print 38/2 flags data structure and indicates the type -t of data to be transmitted. The print server returns this data structure for verification; ppt transmits all the data to the print server and closes the socket.

FILES

printer description file /etc/printcap /usr/spool/* spool directories /usr/spool/*/*3812

contains socket name for 3812 printer

/usr/adm/ppd-errs spooling error log for filter

SEE ALSO

printcap(5), printer3812(5), ibm3812pp(8) "4.3BSD Line Printer Spooler Manual" in UNIX System Manager's Manual IBM 3812 Pageprinter Programming Reference, S544-3268

PSTAT(8)

NAME

pstat - print system facts

SYNOPSIS

/etc/pstat - aixptufT [suboptions] [system] [corefile]

DESCRIPTION

Pstat interprets the contents of certain system tables. If corefile is given, the tables are sought there, otherwise in /dev/kmem. The required namelist is taken from /vmunix unless system is specified. Options are:

- -a Under -p, describe all process slots rather than just active ones.
- -i Print the inode table with these headings:
- LOC The core location of this table entry.
- FLAGS Miscellaneous state variables encoded thus:
 - L locked
 - U update time (fs(5)) must be corrected
 - A access time must be corrected
 - M file system is mounted here
 - W wanted by another process (L flag is on)
 - T contains a text file
 - C changed time must be corrected
 - S shared lock applied
 - E exclusive lock applied
 - Z someone waiting for a lock
- CNT Number of open file-table entries for this inode.
- DEV Major and minor device number of file system in which this inode resides.
- RDC Reference count of shared locks on the inode.
- WRC Reference count of exclusive locks on the inode (this may be > 1 if, for example, a file descriptor is inherited across a fork).
- INO I-number within the device.
- MODE Mode bits, see chmod(2).
- NLK Number of links to this inode.
- UID User ID of owner.

SIZ/DEV

Number of bytes in an ordinary file, or major and minor device of special file.

- -x Print the text table with these headings:
- LOC The core location of this table entry.
- FLAGS Miscellaneous state variables encoded thus:
 - T ptrace(2) in effect
 - W text not yet written on swap device
 - L loading in progress
 - K locked
 - w wanted (L flag is on)
 - P resulted from demand-page-from-inode exec format (see execve(2))
- DADDR Disk address in swap, measured in multiples of 512 bytes.
- CADDR Head of a linked list of loaded processes using this text segment.
- RSS Size of resident text, measured in multiples of 512 bytes.
- SIZE Size of text segment, measured in multiples of 2048 bytes.
- IPTR Core location of corresponding inode.

PSTAT(8)

```
CNT
          Number of processes using this text segment.
CCNT
          Number of processes in core using this text segment.
FORW
          Forward link in free list.
BACK
          Backward link in free list.
          Print process table for active processes with these headings:
— p
LOC
          The core location of this table entry.
S
          Run state encoded thus:
          0
                  no process
          1
                  waiting for some event
          3
                  runnable
          4
                  being created
          5
                  being terminated
          6
                  stopped (by signal or under trace)
F
          Miscellaneous state variables, or'ed together (hexadecimal):
                   loaded
          0001
          0002
                    the scheduler process
          0004
                    locked for swap out
          8000
                    swapped out
                    traced
          0010
          0020
                    used in tracing
          0080
                    in page-wait
          0100
                    prevented from swapping during fork(2)
                    will restore old mask after taking signal
          0200
          0400
                    exiting
          0800
                    doing physical I/O (bio.c)
          1000
                    process resulted from a v fork(2) which is not yet complete
          2000
                    another flag for vfork(2)
          4000
                    process has no virtual memory, as it is a parent in the context of vfork(2)
          8000
                    process is demand paging data pages from its text inode.
          10000
                    process using sequential VM patterns
                    process using random VM patterns
          20000
          100000
                    using old 4.1-compatible signal semantics
                    process needs profiling tick
          200000
          400000
                   process is scanning descriptors during select
          1000000 process page tables have changed
POIP
          number of pages currently being pushed out from this process.
PRI
          Scheduling priority, see setpriority(2).
SIG
          Signals received (signals 1-32 coded in bits 0-31),
UID
          Real user ID.
SLP
          Amount of time process has been blocked.
TIM
          Time resident in seconds; times over 127 coded as 127.
CPU
          Weighted integral of CPU time, for scheduler.
NI
          Nice level, see setpriority(2).
          Process number of root of process group.
PGRP
          The process ID number.
PID
PPID
          The process ID of parent process.
          If in core, the page frame number of the first page of the 'u-area' of the process. If
ADDR
          swapped out, the position in the swap area measured in multiples of 512 bytes.
RSS
          Resident set size — the number of physical page frames allocated to this process.
SRSS
          RSS at last swap (0 if never swapped).
```

SIZE Virtual size of process image (data + stack) in multiples of 2048 bytes. WCHAN Wait channel number of a waiting process. LINK Link pointer in list of runnable processes. **TEXTP** If text is pure, pointer to location of text-table entry. -tPrint table for terminals with these headings: **RAW** Number of characters in raw input queue. CAN Number of characters in canonical input queue. **OUT** Number of characters in output queue. MODE See ttv(4). ADDR Physical device address. Number of delimiters (newlines) in canonical input queue. DEL COL Calculated column position of terminal. **STATE** Miscellaneous state variables encoded thus: T delay timeout in progress W waiting for open to complete \mathbf{O} open F outq has been flushed during DMA \mathbf{C} carrier is on В busy doing output Α process is awaiting output X open for exclusive use S output stopped H hangup on close **PGRP** Process group for which this is controlling terminal. DISC Line discipline; blank is old tty OTTYDISC or "new tty" for NTTYDISC or "net" for NETLDISC (see bk(4)). print information about a user process; the next argument is its address as given by -- u ps(1). The process must be in main memory, or the file used can be a core image and the address 0. Only the fields located in the first page cluster can be located successfully if the process is in main memory. — **f** Print the open file table with these headings: LOC The core location of this table entry. **TYPE** The type of object to which the file table entry points. **FLG** Miscellaneous state variables encoded thus: open for reading R W open for writing open for appending Α S shared lock present X exclusive lock present signal pgrp when data ready **CNT** Number of processes that know this open file. MSG Number of messages outstanding for this file. The location of the inode table entry or socket structure for this file. DATA

- -s print information about swap space usage: the number of (1kb) pages used and free is given, as well as the number of used pages belonging to text images.
- -T prints the number of used and free slots in the several system tables and is useful for checking to see how full system tables have become if the system is under heavy load.

OFFSET The file offset (see lseek(2)).

PSTAT(8)

FILES

/vmunix namelist /dev/kmem default source of tables

SEE ALSO

iostat(1), ps(1), systat(1), vmstat(1), stat(2), fs(5) K. Thompson, UNIX Implementation

BUGS

It would be very useful if the system recorded "maximum occupancy" on the tables reported by —T; even more useful if these tables were dynamically allocated.

PTFINSTALL(8) PTFINSTALL(8)

NAME

ptfinstall - install a Program Temporary Fix (PTF)

SYNOPSIS

/sys/support/ptfinstall [-dinstall_path] ptf#...

DESCRIPTION

This command is for Customer Central Support Site use only. End-users should contact their Customer Central Support Site for assistance in installing a Program Temporary Fix (PTF).

Ptfinstall allows a Customer Central Support Site coordinator to install a PTF. Ptfinstall accepts one or more PTF numbers and installs the corresponding PTF package(s). A PTF package for utility code fixes consists of either source and binary files or only binary files depending on what licenses a site holds. A PTF package for kernel code fixes contains only source code changes.

The options are:

-dinstall path

specifies the base *install_path* directory relative to which the PTFs are installed. The default for *install_path* is /sys/support/ptftest. You can preview fixes in this directory. To install the fixes permanently, specify -d/. You must be superuser to install PTFs permanently.

Because *uucp* limits the size of transmitted packages, a PTF package arrives via the *uucp* support network in pieces. The first time the *ptfinstall* command is issued, the pieces of the PTF package are bundled up into one file and placed in the /sys/support/ptfdir directory. For all subsequent installations of this PTF package the program will look for it in the /sys/support/ptfdir/ptfptf# directory.

FILES

/usr/spool/uucppublic

- directory where the pieces of a PTF package are sent via the uucp support link.

/sys/support/ptfdir

- directory where PTF packages are stored

/sys/support/ptftest

- the default installation directory

SEE ALSO

Support Procedures article in the Program Directory

REBOOT(8) REBOOT(8)

NAME

reboot - bootstrapping procedures

SYNOPSIS

```
/\text{etc/reboot}[-n][-q]
```

DESCRIPTION

IBM/4.3 starts by placing the kernel in memory at location zero and transferring to the entry point. Since the system is not re-enterable, it is necessary to read the kernel in from disk or tape each time it is to be loaded.

Rebooting a running system. When IBM/4.3 is running and multiple users are logged in, shut-down(8) is normally used to perform a reboot. If there are no users, /etc/reboot can be used.

Reboot causes the disks to be synced and allows the system to perform other shutdown activities such as resynchronizing the hardware time-of-day clock. A reboot is then started, as described below. By default, the system boots and the disks are automatically checked. If all this succeeds without incident, the system comes up for multiple users.

Options to reboot are:

- -n do not perform the sync. Use this option if a disk or the processor is on fire.
- -q reboot quickly and ungracefully, without shutting down running processes first.

Reboot normally logs the reboot using syslog(8) and places a shutdown record in the login accounting file /usr/adm/wtmp. These actions are inhibited if the -n or -q options are present.

Power fail and crash recovery. Normally, the system will reboot itself at power-up or after crashes. To force a reboot, simultaneously press and hold the keys labeled < Ctrl>, < Alt> and < Pause>. If all else fails, turn the machine off, wait at least 60 seconds for the disks to stop spinning, and turn it back on.

Boot procedure. After performing diagnostic tests, the system ROM looks for a bootstrap record, searching (in order, and only if present) fd0, fd1, hd0, hd1, and hd2. It loops until it finds one. Normally, the bootstrap record reads the boot program from diskette or hard disk. The boot program prompts for the name of the system to be loaded; the default is hd(0,0)vmunix and takes effect if nothing is typed within about 30 seconds. (Typing a carriage return will also choose the default.) By default, the system comes up multi-user, but if a name was entered, the single-user flag is passed to the kernel.

If the debugger was configured into the kernel, it is entered first, but an implicit "go" will be performed if nothing is entered within about 30 seconds. A carriage return also performs the "go".

FILES

/vmunix system code
/boot system bootstrap

/usr/mdec/xxboot sector-0 boot block for IBM RT PC, xx is specific disk type second-stage boot for IBM RT PC, xx is generic disk type

/usr/mdec/installboot program to install boot blocks on IBM RT PC

SEE ALSO

crash(8R), fsck(8), halt(8), init(8), newfs(8), rc(8), shutdown(8), syslogd(8)

RESTORE(8) RESTORE(8)

NAME

restore - incremental file system restore

SYNOPSIS

/etc/restore key [name ...]

DESCRIPTION

Restore reads tapes dumped with the dump(8) command. 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 the files that are to be restored. Unless the h key is specified (see below), the appearance of a directory name refers to the files and (recursively) subdirectories of that directory.

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

r The tape is read and loaded into the current directory. This should not be done lightly; the r key should only be used to restore a complete dump tape onto a clear file system or to restore an incremental dump tape after a full level zero restore. Thus

```
/etc/newfs /dev/rrp0g eagle
/etc/mount /dev/rp0g /mnt
cd /mnt
restore r
```

is a typical sequence to restore a complete dump. Another restore can be done to get an incremental dump in on top of this. Note that restore leaves a file restoresymtab in the root directory to pass information between incremental restore passes. This file should be removed when the last incremental tape has been restored.

A dump(8) followed by a newfs(8) and a restore is used to change the size of a file system.

- R Restore requests a particular tape of a multi volume set on which to restart a full restore (see the r key above). This allows restore to be interrupted and then restarted.
- x The named files are extracted from the tape. If the named file matches a directory whose contents had been written onto the tape, and the h key is not specified, the directory is recursively extracted. The owner, modification time, and mode are restored (if possible). If no file argument is given, then the root directory is extracted, which results in the entire content of the tape being extracted, unless the h key has been specified.
- t The names of the specified files are listed if they occur on the tape. If no file argument is given, then the root directory is listed, which results in the entire content of the tape being listed, unless the h key has been specified. Note that the t key replaces the function of the old dumpdir program.
- This mode allows interactive restoration of files from a dump tape. After reading in the directory information from the tape, *restore* provides a shell like interface that allows the user to move around the directory tree selecting files to be extracted. The available commands are given below; for those commands that require an argument, the default is the current directory.
 - Is [arg] List the current or specified directory. Entries that are directories are appended with a "/". Entries that have been marked for extraction are prepended with a "*". If the verbose key is set the inode number of each entry is also listed.
 - cd arg Change the current working directory to the specified argument.
 - pwd Print the full pathname of the current working directory.
 - add [arg] The current directory or specified argument is added to the list of files to be

RESTORE(8) RESTORE(8)

extracted. If a directory is specified, then it and all its descendents are added to the extraction list (unless the h key is specified on the command line). Files that are on the extraction list are prepended with a "*" when they are listed by Is.

- delete [arg] The current directory or specified argument is deleted from the list of files to be extracted. If a directory is specified, then it and all its descendents are deleted from the extraction list (unless the h key is specified on the command line). The most expedient way to extract most of the files from a directory is to add the directory to the extraction list and then delete those files that are not needed.
- extract All the files that are on the extraction list are extracted from the dump tape.

 Restore will ask which volume the user wishes to mount. The fastest way to extract a few files is to start with the last volume, and work towards the first volume.
- verbose The sense of the v key is toggled. When set, the verbose key causes the ls command to list the inode numbers of all entries. It also causes restore to print out information about each file as it is extracted.
- help List a summary of the available commands.
- quit Restore immediately exits, even if the extraction list is not empty.

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

- v Normally restore does its work silently. The v (verbose) key causes it to type the name of each file it treats preceded by its file type.
- The next argument to restore is used as the name of the archive instead of /dev/rmt?. If the name of the file is "-", restore reads from standard input. Thus, dump(8) and restore can be used in a pipeline to dump and restore a file system with the command

```
dump 0f - /usr | (cd /mnt; restore xf -)
```

F The next argument to restore is used as the name of a file from which interactive input is read. Normally, standard input (or the controlling terminal if the f key specifies standard input) is read. This flag allows the interactive mode of restore to be driven from a command file when the archive file is standard input. The interactive interface, the prompt for next volume number, and the prompt to set the access mode for "." are affected. Error recovery interaction and verifying operator readiness are not affected. For example, if the file inputfile contains

```
add
delete foo
add foo/bar
extract
l
yes
quit
```

then the command

restore iF inputfile

will use the interactive mode to automatically mark everything for extraction, unmark the directory foo, mark foo/bar, extract the marked files, specify volume 1, set the access mode for ".", and quit. The easiest way to determine the commands needed is to do the restore by hand once, and write down everything that you type.

RESTORE(8) RESTORE(8)

y Restore will not ask whether it should abort the restore if gets a tape error. It will always try to skip over the bad tape block(s) and continue as best it can.

- m Restore will extract by inode numbers rather than by file name. This is useful if only a few files are being extracted, and one wants to avoid regenerating the complete pathname to the file.
- h Restore extracts the actual directory, rather than the files that it references. This prevents hierarchical restoration of complete subtrees from the tape.

DIAGNOSTICS

Complaints about bad key characters.

Complaints if it gets a read error. If y has been specified, or the user responds "y", restore will attempt to continue the restore.

If the dump extends over more than one tape, restore will ask the user to change tapes. If the x or i key has been specified, restore will also ask which volume the user wishes to mount. The fastest way to extract a few files is to start with the last volume, and work towards the first volume.

There are numerous consistency checks that can be listed by restore. Most checks are self-explanatory or can "never happen". Common errors are given below.

Converting to new file system format.

A dump tape created from the old file system has been loaded. It is automatically converted to the new file system format.

< filename >: not found on tape

The specified file name was listed in the tape directory, but was not found on the tape. This is caused by tape read errors while looking for the file, and from using a dump tape created on an active file system.

expected next file < inumber >, got < inumber >

A file that was not listed in the directory showed up. This can occur when using a dump tape created on an active file system.

Incremental tape too low

When doing incremental restore, a tape that was written before the previous incremental tape, or that has too low an incremental level has been loaded.

Incremental tape too high

When doing incremental restore, a tape that does not begin its coverage where the previous incremental tape left off, or that has too high an incremental level has been loaded.

Tape read error while restoring < filename >

Tape read error while skipping over inode < inumber >

Tape read error while trying to resynchronize

A tape read error has occurred. If a file name is specified, then its contents are probably partially wrong. If an inode is being skipped or the tape is trying to resynchronize, then no extracted files have been corrupted, though files may not be found on the tape.

resync restore, skipped < num > blocks

After a tape read error, *restore* may have to resynchronize itself. This message lists the number of blocks that were skipped over.

FILES

/dev/rmt? the default tape drive

/tmp/rstdir* file containing directories on the tape.

/tmp/rstmode* owner, mode, and time stamps for directories.

./restoresymtab information passed between incremental restores.

RESTORE(8)

SEE ALSO

rrestore(8C) dump(8), newfs(8), mount(8), mkfs(8)

BUGS

Restore can get confused when doing incremental restores from dump tapes that were made on active file systems.

A level zero dump must be done after a full restore. Because restore runs in user code, it has no control over inode allocation; thus a full dump must be done to get a new set of directories reflecting the new inode numbering, even though the contents of the files is unchanged.

NAME

restore.tape, restore.net - install system from tape or over network

SYNOPSIS

restore.tape [options] partition ... restore.net [options] partition ...

DESCRIPTION

Note: These commands are only on the installation miniroot.

Restore.tape installs a system from distribution tapes. Restore.net installs a system from one host to another over a local area network. Both use restore(8) and read a dump(8) archive, either from the streaming tape or from dump(8) executed on a remote host via rsh(1C).

Restore.tape and restore.net are used from the miniroot diskette. Normally, they are not used directly, although they can be used directly to perform non-standard installations.

The following options are available to both restore.tape and restore.net:

-T hdxv

Mount the writable /tmp directory on the y partition of the hdx disk. If y is omitted, it defaults to b. If the -T option is omitted, hd0b is used.

-fn The file /etc/fstab.hd.n is copied into /etc/fstab on the root partition after it is installed. N may be 1 or 3. The original /etc/fstab is copied into /etc/fstab.old. Without the -fn option, /etc/fstab is unchanged. These options work only if the root partition is installed as hd?? = root or as hd? = root/user.

-k kernel

Install /kernel into /vmunix. For example, -k vmunix.special will install /vmunix.special into /vmunix after the root partition has been installed. This option works only if the root partition is installed as hd?? = root or as hd? = root/user.

-m mask

Specify a *mask* which prevents certain parts of the system from being installed. Currently available masks are:

nodoc Prevents installation of documentation source in /usr/doc.

notont Prevents installation of the 3812 font libraries in /usr/lib/font.

nokernel

Prevents installation of kernel source in /usr/sys, except for header files normally available via symbolic links in /usr/include.

nolearn Prevents installation of the learn(1) data base library in /usr/lib/learn.

noman Prevents installation of on-line man pages in /usr/man.

nonotes

Prevents installation of notesfiles in /usr/spool/notes.

nosupport

Prevents installation of site support tools in /usr/sys/support.

nouser Prevents installation of the contents of the user partition when installing the root partition. This is useful for installing just the root partition from the ROOT/USER tape. See the **EXAMPLES** section below.

- newfs string

Pass string to newfs on the command line. By default, the string "-m 5" is passed to newfs.

- -q Ouiet. Reduces the volume of output to the screen.
- -x The command newfs(8) is not used to create new filesystems. See the BUGS section below.

The following option is available only to restore.tape:

-bn Use n thuffer (8R) buffers to buffer streaming tape. If n is omitted, or if the -bn option is not used, 4 buffers are used. If n is 0, thuffer (8R) is not used.

The following options are available only to restore.net:

-s sourcehost

Use sourcehost as the remote host for installation. Sourcehost may be either a hostname (ie, "master") or an internet address (ie, "46.0.0.1"), and defaults to master. Sourcehost must have an entry for desthost in its /etc/hosts, /etc/hosts.equiv, and /.rhosts files.

-d desthost

Configure network interface as desthost. Desthost may be either a hostname (ie, "slave") or an internet address (ie, "46.0.0.2"), and defaults to slave.

— un

-- lan

Specify preference for un0 (ethernet) or lan0 (token ring) network interface. By default, restore.net attempts to configure un0, then lan0. If the -lan option is specified, lan0 is tried first, then un0.

- ifconfig string

Pass string to ifconfig on the command line. By default, the null string is passed to ifconfig. This allows users to install over networks using protocols other than the default. See ifconfig(8) if your site does not use the default protocols.

The allowed partitions for both restore.tape and restore.netare:

disk Install disk from tape, or from disk on master host. Disk may be hd[0-2][a-h] or sc[01][a-h].

disk = name

Install disk fromtape, or from the named file system on master host. Name may be root, user, or source. Note that mastking (-m mast) is done based on name, and will NOT be done unless this format is used.

The allowed partitions for restore.tape only are:

disk

disk = name

For tape installation, disk may also be hd[0-2], as before. Name must be root/user. This is used for installing distribution root/user tapes.

The allowed partitions for restore.net only are:

disk1 = disk2

The contents of disk2 on the master host are installed on disk1 on this host. Note that masking is not done!

disk = filesystem

Filesystem may be any mountpoint listed in /etc/fstab on the master host. It does the same thing as the above format would if disk2 were the partition mounted on the mountpoint filesystem. Note that masking is not done!

Note that the following partition formats are no longer allowed:

disk:name

Use disk = name instead.

disk = disk:name

Use disk = name or disk1 = disk2 instead.

EXAMPLES

Install only the root partition from the ROOT/USER tape:

restore.tape -m nouser hd0a = root

Install all distribution tapes normally:

restore.tape -f3 hd0 = root/user hd1g = source

Install the hd2c partition from the hd2g partition on myhost:

restore.tape -s myhost hd2c = hd2g

BUGS

Using the -x option to prevent the use of newfs(8) is restricted in overlaying a new system onto local additions, because restore(8) cannot create new hard links to new files, when the old hard links to the old files still exist. Symbolic links are affected the same way, although usually they are not as critical.

The character "=" is not allowed in name (disk = name) or file system (disk = filesystem).

RVDCHLOG(8) RVDCHLOG(8)

NAME

rvdchlog - change logging level of Remote Virtual Disk (RVD) server

SYNOPSIS

/usr/ibm/rvdchlog[-d] server loglevel

DESCRIPTION

Rvdchlog will change the logging level of the RVD server on the host server. The $-\mathbf{d}$ flag enables debugging. For a description of what loglevel means, see rvdsrv(8).

SEE ALSO

rvdlog(8), rvdsrv(8)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

RVDCOPY(8)

NAME

rvdcopy - copy contents of one RVD disk pack to another

SYNOPSIS

/usr/ibm/rvdcopy from_drive to_drive [blockcount [startblock]]

DESCRIPTION

Rvdcopy causes the contents of one RVD disk pack to be copied to another. From_drive and to_drive are single digit integers of the drive numbers to be copied from and to. Blockcount is the number of blocks to be copied and startblock says where to start copy from.

SEE ALSO

rvdtab(5), savervd(8)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

RVDDOWN(8) RVDDOWN(8)

NAME

rvddown - force spindown of a Remote Virtual Disk (RVD) pack

SYNOPSIS

/usr/ibm/rvddown server pack

DESCRIPTION

Rvddown will force spindown of all connections involving the pack pack on the server host server. The exclusive mode password for pack is required for this operation.

Warning: This operation should be used with care; if a client system has the pack spun up when the pack is shut down it will crash. Also, an exclusive mode pack may have a corrupted filesystem should this happen.

SEE ALSO

rvdshut(8), rvdflush(8), rvdsrv(8)

RVDEXCH(8)

NAME

rvdexch - exchange names of two Remote Virtual Disk (RVD) packs

SYNOPSIS

/usr/ibm/rvdexch [-d] server pack1 uid1 pack2 uid2

DESCRIPTION

Rvdexch exchanges the names of two RVD packs on a server host server. It will prompt for exclusive-mode passwords of each pack, and then send a name exchange request to the RVD server. The $-\mathbf{d}$ flag enables debugging. Pack1 and pack2 are names of packs and must be presently associated with the two packs. Uid1 and uid2 are unique identifiers of the packs as given in |etc|rvd|rvddb.

Note: The nature of the change instilled by *rvdexch* is ephemeral in that it exists only on the running server, and will disappear when the server is next restarted.

BUGS

The current control protocol provides no fail-safe mechanism in the event of a lost acknowledgement from the server. Should this happen, another request would be sent, undoing the effects of the previous one. This may be circumvented by running *rvdexch* on the server machine, thereby eliminating the possibility of a lost acknowledgement.

SEE ALSO

rvddb(5), rvdsrv(8)

RVDFLUSH(8) RVDFLUSH(8)

NAME

rvdflush - spindown client's Remote Virtual Disk (RVD) packs

SYNOPSIS

/usr/ibm/rvdflush[-d] server client

DESCRIPTION

In executing rvdflush, which is typically done at boot-time, all potential lingering connections to server from client are flushed (shut-down). See the use of rvdflush in |etc|rvd|rvdstart script for more details.

Warning: This should be done only for another client when it fails to flush itself, or has crashed. In either form, the $-\mathbf{d}$ flag enables debugging.

SEE ALSO

rvdtab(5), rvddown(8), rvdshut(8)

RVDGETM(8)

NAME

rvdgetm - get operations message from Remote Virtual Disk (RVD) server

SYNOPSIS

/usr/ibm/rvdgetm [-d] [server]

DESCRIPTION

Rvdgetm gets the operations message from the RVD server host server, or if none is specified, from each server on which a drive is spun-up (see vdstats(8)). The -d flag enables debugging. The message (if any) is labeled with the originating server's hostname. The purpose of these messages is to notify users of changes in RVD service.

SEE ALSO

up(1), rvdsetm(8), rvdsrv(8)

RVDHOSTS(8) RVDHOSTS(8)

NAME

rvdhosts - print a list of RVD servers

SYNOPSIS

/usr/ibm/rvdhosts

DESCRIPTION

Rvdhosts reads an /etc/rvd/rvdtab file and prints a list of RVD servers listed there.

SEE ALSO

rvdtab(5)

RVDLOG(8)

NAME

rvdlog - cause Remote Virtual Disk (RVD) server to log statistics

SYNOPSIS

/usr/ibm/rvdlog [-d][-a] server

DESCRIPTION

Rvdlog causes the RVD server on the host server to dump statistics to its log file. The $-\mathbf{d}$ flag enables debugging. The $-\mathbf{a}$ (all) flag causes a more exhaustive statistics dump.

SEE ALSO

rvdchlog(8), rvdsrv(8), syslogd(8)

RVDSEND(8) RVDSEND(8)

NAME

rvdsend - send control stream to Remote Virtual Disk (RVD) server

SYNOPSIS

/usr/ibm/rvdsend [-d] [[server] file]

DESCRIPTION

Rvdsend reads control strings as described in Chapter 5, "RVD Control Protocol Specification," in "The Remote Virtual Disk System" in Volume II, Supplementary Documents, from file (default standard input), and sends them to the server on the host server (default local host). This is usually used for initializing the server from the master disk layout file at boot time.

Note that the contents of the file are not restricted, so *rvdsend* may be used for sending arbitrary requests, as in debugging. Debugging may be enabled with the $-\mathbf{d}$ flag.

SEE ALSO

rvddb(8), rvdsrv(8)

RVDSETM(8) RVDSETM(8)

NAME

rvdsetm - set operations message on Remote Virtual Disk (RVD) server

SYNOPSIS

/usr/ibm/rvdsetm[-d][-r][-f] server [message]

DESCRIPTION

Rvdsetm sets the operations message on the RVD server host server. The $-\mathbf{d}$ flag enables debugging. If the flag $-\mathbf{r}$ is given, the message is cleared; otherwise, the message is taken as the argument (if supplied), or read from standard input. The $-\mathbf{f}$ flag inhibits prompting for the operations password. This is useful when running rvdsetm in a script, before a "require_authorization" operation has been performed on the server.

The purpose of these messages is to notify users of changes in RVD service. A sample message might be:

New release of /usr tonight.

Reboot your machine to get the latest and greatest!

or

Jinx going down tonight 20h00 to 21h00 for preventative maintenance.

SEE ALSO

up(1), rvdgetm(8), rvdsend(8), rvdsrv(8)

RVDSHOW(8) RVDSHOW(8)

NAME

rvdshow - show connections to Remote Virtual Disk (RVD) server

SYNOPSIS

```
/usr/ibm/rvdshow [ -d ] server [ pack ]
/usr/ibm/rvdshow [ -d ] server [ -c client ]
```

DESCRIPTION

Rvdshow lists information regarding connections to the RVD server host server. If pack is specified, connections are listed involving that disk. If client is specified, connections initiated by this host to server are listed. In either form, the $-\mathbf{d}$ flag enables debugging.

The information displayed includes pack names, their current spinup mode, and connections. If pack or client is specified, connections are given as client host and drive number; otherwise, it is summarized as connection count and time since last activity.

SEE ALSO

up(1), rvdsrv(8), spinup(8)

RVDSHUT(8)

NAME

rvdshut - force shutdown of Remote Virtual Disk (RVD) server

SYNOPSIS

/usr/ibm/rvdshut[-d][server]

DESCRIPTION

Rvdshut will shut down the RVD server on the host server (default local host). It prompts at the terminal for the operations password. The $-\mathbf{d}$ flag enables debugging.

Warning: This should only be used after all disk packs have been spun down. Any pack spun up when the RVD server is shut down may cause the client system to crash when it next tries to access that disk pack.

SEE ALSO

rvddown(8), rvdflush(8), rvdsrv(8)

RVDSRV(8) RVDSRV(8)

NAME

```
rvdsrv - Remote Virtual Disk (RVD) server daemon
```

SYNOPSIS

```
/usr/ibm/rvdsrv [ - I loglevel ] [ - r [restart file] ]
```

DESCRIPTION

Rvdsrv is the RVD server program which handles the physical media and services read and write requests from network connections. It also accepts control requests on a UDP port to perform other activities such as spin up or down a disk, create a virtual disk, or add a physical device. It logs actions into the syslog (see syslogd(8)).

Loglevel is compared against the following log levels:

```
#define LOG_ERR 1 /* log all packet errors */
#define LOG_SPINS 2 /* log all spinups/spindown */
#define LOG_RDWR 4 /* log all read and write requests */
#define LOG_CLIENT_ERROR 8 /* log all client errors */
#define LOG_TRACE 16 /* log a full packet trace */
```

If a bit in *loglevel* is on, the corresponding logging occurs. For example, if *loglevel* is 5, read and write requests will be logged, along with packet errors.

The -r switch will cause the server to restart if it encounters a fatal error. During a crash the server will do a core dump and read commands from the default or given restart_file before aborting.

DIAGNOSTICS

Diagnostics are handled by syslog(8).

FILES

```
/etc/rvd/rvddb database of current disk layout
/etc/services port assignments
/etc/rvd/rvdstart default restart file
```

SEE ALSO

rvddb(5), vddb(8), syslogd(8)

SAUTIL(8R) SAUTIL(8R)

NAME

sautil - standalone utility package

DESCRIPTION

Sautil is a package of standalone utility programs used in maintaining IBM/4.3. The package contains the following utilities:

Boot is a standalone program used to boot the kernel (vmunix) into memory and start it executing.

Format is a standalone program used to format and check disks prior to constructing file systems.

Dump makes a hexadecimal display (or dump) of a specified disk or file. It can also be used to make changes to a disk (useful for emergency repair of the disk configuration record at sector 1).

Cat displays an ASCII file on a screen (similar to cat(1) utility).

Ls displays the contents of a directory (similar to ls(1) utility). This is useful for cases where the primary |vmunix| has been damaged and a backup copy is known to exist somewhere in the filesystem tree. By repeatedly using ls, one can search the directory tree for a given file.

Copy makes a copy of all or part of a disk or diskette. One use of this is to restore an overwritten bad-block table to the start of the disk (starting at block 8) from a backup copy in another filesystem or from a diskette.

Debug is the debugger (see debug(8)). It may be used to display memory or access I/O devices.

Iplsource changes the IPL (Initial Program Load) or boot order in the non-volatile ram. This is used to select the order in which devices are checked for a valid boot block. The default is to first check the diskettes and then the hard disks.

Minidisk displays and allows changes to be made to the minidisk directory. The directory is in VRM compatible format. It can be used to change the size and location of the hard disk partitions.

Dosboot allows a program to be booted off a DOS-format diskette. This allows several standalone utilities to be present on one diskette.

Convert is actually a special way of invoking format(8R). It should be used only when installing the Extended ESDI disk adapter.

Input to these utilities may be edited using the standard line editing characters "^H" and "^U".

For the IBM 6152 Academic System, fdisk displays and allows changes to be made to the master boot record partition table. The table format is described under "Fixed Disk Information" in the DOS 3.3 Reference Manual.

EXAMPLE

In this example, sautil is loaded from a diskette; it can also be loaded from the stand directory in the |usr(hd(0,6))| file system (e.g. hd(0,6)stand/sautil). Boldface indicates user input.

4.3 BSD UNIX Standalone Maintenance Program \$Revision: 9.3 \$

Choice

Description

- 1 boot boot standalone program or kernel
- 2 format format hard disk
- 3 dump display disk or diskette (hex)
- 4 cat display a file contents (ASCII)
- 5 ls print directory of filesystem
- 6 copy copy all/part of disk or diskette

```
7 debugger - display memory, etc.
8 iplsource - set boot order in nvram
9 minidisk - display/change minidisk directory
```

10 dosboot - boot standalone program or kernel from DOS diskette

11 convert - R70 ("hd70r") disk to an E70

Enter the menu choice number desired, then press < Enter >

Choice: 5

```
Directory: hd(0,0).
2
2
4
                .profile
768
                bin
5
                boot
1163
                dev
384
                etc
1152
                lib
                lost + found
3
2304
                mnt
6
                sys
16
                tmp
1920
                usr
11
                vmunix
Directory:
```

NOTES

Sautil is packaged as the boot (8) program in |boot and on the installation diskette. The sautil menu is not presented unless the boot program exits, which may be caused by typing control-C when input is expected (at the reboot prompt). Not all of sautil's utilities are available when packaged as boot due to space limitations.

If the utility invokes the exit routine (either as a normal exit or as the result of an error), the message

Exit called - Press Enter to return to main menu

will be displayed. Pressing the < Enter > key displays the sautil menu.

SEE ALSO

boot(8), debug(8), format(8R), minidisk(8R),

BUGS

The selection of utilities in *sautil* is somewhat limited; however, experience has shown that this set is sufficient for most purposes. If a richer set of utilities is required, it is usually best to use a diskette-based miniroot with the normal utilities running on a diskette-based kernel.

SAVERVD(8) SAVERVD(8)

NAME

savervd, zaprvd, savephys - back up and restore Remote Virtual Disk (RVD) packs to and from tape

SYNOPSIS

```
/usr/ibm/savervd [ -f tape ] [ -m database ] pack1 ...
/usr/ibm/savephys [ -f tape ] [ -m database ] phys1 ...
/usr/ibm/zaprvd [ -f tape ] [ -m database ] pack1 ...
```

DESCRIPTION

Savervd saves virtual disks pack!... to device tape, default |dev|rmt8. Savephys saves all virtual disks on physical devices phys!... to tape. Zaprvd restores saved virtual disks pack!... from device tape.

The virtual disks must already exist and be named in database, default /etc/rvd/rvddb (see rvddb(5)).

If the user is not the superuser, the read-only password must be supplied to save the RVD packs or the exclusive password to restore them.

Note that when saving or restoring more than one pack, the non-rewinding device is to be used (e.g. use |dev|nrst0 and not |dev|rst0 in case of using a streaming-tape).

SEE ALSO

rvddb(5)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

BUGS

The command names aren't particularly consistent.

The default tape is inappropriate to the IBM RT PC.

SCSIFORMAT(8C) SCSIFORMAT(8C)

NAME

scsiformat - format the IBM 9332 disk unit

SYNOPSIS

scsiformat device level [interleave]

DESCRIPTION

The scsiformat program writes formatting information on the IBM 9332 disk units specified by device. Device must be a raw device (/dev/rscsi??) for one of the partitions of the disk you wish to format. The whole disk unit is formatted, regardless of the partition specified.

Level is an integer between 1 and 3 inclusive specifying how the disk is to be formatted:

- 1 Change the interleave factor only (don't actually format the drive).
- 2 Use only the manufacturer's defect table that came with the disk unit to format.
- 3 Use both the manufacturer's defect table and the "grown" defect table (the defect table generated by the disk unit itself as it discovers bad blocks in its normal operation) to format.

Interleave is the sector interleave factor to use. This can be changed without actually formatting the disk. Legal values for interleave are integers between 0 and 8 inclusive. An interleave of 0 specifies the default interleave value set in the drive at the time it was shipped. If interleave is not specified, an interleave of 0 is assumed.

Scsiformat does not serve the same purpose as the hd formatter, since the drive will automatically start forwarding bad blocks it detects while running, without reformatting. Scsiformat does not attempt to preserve data; using scsiformat will destroy any data currently on the disk. Because of this, scsiformat will prompt you to verify the operation you specified in level.

FILES

/dev/rscsi [0-13] [a-h]

SEE ALSO

scsi(4)

BUGS

Use of the other scsi devices is suspended while scsiformat is running.

SENDAPAR(8) SENDAPAR(8)

NAME

sendapar - send APAR

SYNOPSIS

/sys/support/sendapar

DESCRIPTION

Note: This command is intended for Customer Central Support Site use only. Only a superuser or member of the group "staff" can use this command. End-users should contact their Customer Central Support Site for assistance before submitting an Authorized Program Analysis Report (APAR), i.e., reporting a problem.

Sendapar allows a Customer Central Support Site coordinator to report problems by submitting an Authorized Program Analysis Report (APAR). The APAR is forwarded to IBM Academic Information Systems (ACIS).

To submit an APAR, type sendapar. A template will appear on the display. Complete the template using vi(1) (by default) or whatever editor is specified by the environment variable EDITOR. The strings "%%" and "%@" indicate where to supply information. Some fields are filled in by sendapar, some fields are optional, and others must be completed. Type over the leading percent on any line on which you supply information. When the template is filled in as much as possible, issue a :wq for vi, or the appropriate write-quit sequence for your editor. Blank lines and any lines containing a "%@" are removed, and the APAR is sent automatically to IBM ACIS. To cancel a sendapar report, quit the editing session without writing changes to the template file. Unchanged APAR templates are not forwarded.

FILES

/sys/support/lib - utility programs reside here /sys/support/lib/BUGNUMAPAR - current report number

SEE ALSO

Support Procedures article in the Program Directory

SETSCREEN(8) SETSCREEN(8)

NAME

setscreen - control display screen access

SYNOPSIS

/etc/setscreen mode name /etc/setscreen -f [file]

DESCRIPTION

Setscreen is useful in preventing inadvertent (or deliberate) manipulation of the screen contents on a given screen when using multiple display devices. Setscreen takes an access mode and a device name from its command line and instructs the kernel to allow or prevent access to the display device indicated. Alternatively, the modes and names can be in a file specified by a -f option. If file is omitted, /etc/consoles is used.

Values for mode and name are described in consoles(5).

FILES

/etc/consoles

SEE ALSO

cons(4), consoles(5)

SETID(8)

NAME

setid - set user and group IDs

SYNOPSIS

```
/etc/setid -f file [ args ... ]
/etc/setid [user].[group] command [ args ... ]
```

DESCRIPTION

Setid sets the real and effective user and group IDs to the specified values, and executes a command. If the user field is not specified, then it is reset to whatever the value of the real user ID is.

If the -f flag is specified, then the user and group arguments are read in from the first line of file. Similarly, command is read in from the second line of file.

Normally setid is never used from the command line. Its purpose is to allow setuid-to-root "front-end" shell scripts to call other programs with specific effective user and group IDs. This is useful in the Andrew File System, where set-uid works only for root, and set-gid does not work at all

EXAMPLE

Here is a typical "front-end" shell script that allows setgid to work in the Andrew File System:

```
#! /etc/setid -f
root.daemon
/andrew/usr/ucb/.lpq
```

If an executable set-uid-to-root shell script like this is in /andrew/usr/ucb/lpq, then /andrew/usr/ucb/.lpq (the original lpq program) will be called with effective user and group IDs of root and daemon, respectively. If you use Loadafs(1V), shell scripts of this sort will be set up automatically.

SEE ALSO

Loadafs(1V)

SPINUP(8)

NAME

spinup, spindown - spin up/down Remote Virtual Disk (RVD) pack

SYNOPSIS

```
spinup [-f] drive server name mode [ password ] spindown -a spindown drive
```

DESCRIPTION

Spinup spins up an RVD pack. The -f flag forces prompting for the command authorization password. Drive is a single digit integer denoting the drive number on which to spin up the virtual pack. Server is the name of the host serving the pack. Name is the name of the remote disk pack. Mode is the mode in which to spin up the virtual pack. Modes may be one of only 'r' or 'x' for either read-only or exclusive mode. Password is the password for the specific spinup mode. Password is unnecessary if the local host "owns" the remote pack. If the pack is not owned by the local host and the password is not supplied on the command line, spinup prompts for the password.

Spinup changes ownership of the devices associated with drive to that of the invoking user (see getuid(2)).

Spindown spins down an RVD pack. The -a flag spins down all drives. Drive is a single digit integer denoting the drive number to spin down.

Spindown restores the ownership of the devices associated with drive to root.

Warning: The virtual file system must be unmounted when the virtual disk is spun down, or the system may hang or crash (see *umount*(1)).

FILES

```
/dev/vd[0-9]a block special file pseudo-device
/dev/rvd[0-9]a character special file pseudo-device
```

SEE ALSO

```
chown(1), up (1), getuid(2), rvdspin(8)
"The Remote Virtual Disk System" in Volume II, Supplementary Documents
```

SYSCALL(8)

NAME

syscall - system call interface program

SYNOPSIS

/etc/syscall [-n] name [arg ...]

DESCRIPTION

Syscall is used to reduce the size of a miniroot by replacing several large, commonly used utilities with small shell scripts that call one small program (syscall) in order to perform system calls. If n is specified, syscall performs the system call(s) n times. No error checking is done on the arguments passed to a system call.

The arguments can be of the following formats:

; Separates multiple system calls (up to a maximum of 20) issued by the same invocation of syscall.

0xnnn Hexadecimal constant nnn.

0nnn Octal constant nnn.

nnn

+ nnn

-nnn

Decimal constant nnn.

"string

'string

\string

The character string "string".

#string The length of the character string "string".

&&n The address of the *n*th argument to this system call (n=0) is the system call name).

& n The address of the nth byte in an internal 10k buffer.

Sn The result of the *n*th system call (n = 0) is the first system call).

string Anything else is a literal character string.

Syscall also understands the following special case "system calls":

sleep The library function sleep(3).

EXAMPLE

```
The C fragment:
```

```
output = open("x",1);
write(output,"hello",strlen("hello"));
```

can be simulated by:

```
syscall open x 1 \; write \$0 hello \#hello
```

Note that characters that are special to the shell must be escaped. Refer to sh(1) or csh(1), depending on which shell you are using.

DIAGNOSTICS

Prints message and exits for unknown system calls and for system calls that return -1.

SEE ALSO

```
intro(2), sleep(3), syscall(2)
```

SYSCALL(8)

BUGS

Syscall does no error checking of its arguments.

SYSLOGD(8) SYSLOGD(8)

NAME

syslogd - log systems messages

SYNOPSIS

```
/etc/syslogd [ -fconfigfile ] [ -mmarkinterval ] [ -d ]
```

DESCRIPTION

Syslogd reads and logs messages into a set of files described by the configuration file |etc| syslog.conf. Each message is one line. A message can contain a priority code, marked by a number in angle braces at the beginning of the line. Priorities are defined in $\langle sys/syslog.h \rangle$. Syslogd reads from the UNIX domain socket |dev|log, from an Internet domain socket specified in |etc| services, and from the special device |dev|klog (to read kernel messages).

Syslogd configures when it starts up and whenever it receives a hangup signal. Lines in the configuration file have a selector to determine the message priorities to which the line applies and an action. The action field are separated from the selector by one or more tabs.

Selectors are semicolon separated lists of priority specifiers. Each priority has a facility describing the part of the system that generated the message, a dot, and a level indicating the severity of the message. Symbolic names may be used. An asterisk selects all facilities. All messages of the specified level or higher (greater severity) are selected. More than one facility may be selected using commas to separate them. For example:

*.emerg;mail,daemon.crit

Selects all facilities at the emerg level and the mail and daemon facilities at the crit level.

Known facilities and levels recognized by syslogd are those listed in syslog(3) without the leading "LOG_". The additional facility "mark" has a message at priority LOG_INFO sent to it every 20 minutes (this may be changed with the -m flag). The "mark" facility is not enabled by a facility field containing an asterisk. The level "none" may be used to disable a particular facility. For example,

*.debug;mail.none

Sends all messages except mail messages to the selected file.

The second part of each line describes where the message is to be logged if this line is selected. There are four forms:

- A filename (beginning with a leading slash). The file will be opened in append mode.
- A hostname preceded by an at sign ("@"). Selected messages are forwarded to the syslogd on the named host.
- A comma separated list of users. Selected messages are written to those users if they are logged in.
- An asterisk. Selected messages are written to all logged-in users.

Blank lines and lines beginning with '#' are ignored.

For example, the configuration file:

kern,mark.debug
*.notice;mail.info
*.crit /usr/adm/critical
kern.err
*.emerg
*.alert eric,kridle
*.alert;auth.warning
/dev/console
/usr/spool/adm/syslog
/usr/adm/critical

SYSLOGD(8) SYSLOGD(8)

logs all kernel messages and 20 minute marks onto the system console, all notice (or higher) level messages and all mail system messages except debug messages into the file /usr/spool/adm/syslog, and all critical messages into /usr/adm/critical; kernel messages of error severity or higher are forwarded to ucbarpa. All users will be informed of any emergency messages, the users "eric" and "kridle" will be informed of any alert messages, and the user "ralph" will be informed of any alert message, or any warning message (or higher) from the authorization system.

The flags are:

- f Specify an alternate configuration file.
- -m Select the number of minutes between mark messages.
- -d Turn on debugging.

Syslogd creates the file /etc/syslog.pid, if possible, containing a single line with its process ID. This can be used to kill or reconfigure syslogd.

To bring syslogd down, it should be sent a terminate signal (e.g. kill 'cat /etc/syslog.pid').

NOTES

If a facility is set more than once on a single line, the rightmost setting will be used. This is particularly important when using asterisks, which should normally be placed lestmost on a line. For example:

auth.notice; *.err

will select all facilities at the err level (auth, too!), whereas:

*.err:auth.notice

will select the auth facility at the notice level, and all other facilities at the err level.

Syslogd will not notice duplicate action fields. For example:

*.eri

/usr/spool/adm/syslog

auth.notice

/usr/spool/adm/syslog

will cause all error messages (or higher) from the authorization system to be written to /usr/spool/adm/syslog twice. Also, the file /usr/spool/adm/syslog will be opened twice. Unless such behavior is desired, the above example should be combined into one line, as:

*.err; auth.notice

/usr/spool/adm/syslog

FILES

/etc/syslog.conf the configuration file

/etc/syslog.pid

the process id

/dev/log

Name of the UNIX domain datagram log socket

/dev/klog

The kernel log device

SEE ALSO

logger(1), syslog(3)

TAILOR(8) TAILOR(8)

NAME

tailor - work station customizing assistance

SYNOPSIS

/etc/tailor

DESCRIPTION

Tailor helps you customize a work station after you have installed the 4.3/RT software. A work station can be an end-user, a master for network installation, or a uucp connection machine. For an end-user or master type the command prompts for a hostname. For a uucp connection machine the command prompts for a hostname, serial port, modern type, password and the IBM ACIS telephone number. In the tailor script or help files indicate what choices are available or how to get the necessary information.

FILES

/usr/lib/tailor/help.*

/usr/lib/tailor/tailor.*

/etc/hosts

/etc/rc.local

/etc/ttys

/usr/lib/uucp/L.sys

/usr/lib/uucp/L-devices

- help files

- auxiliary scripts

- host name data base

- system restart script

- terminal configuration data

- host data for uucp

- call-unit data for uucp

SEE ALSO

hostname(1)

Support Procedures article in the Program Directory

TAILOR(8)

TBUFFER(8) TBUFFER(8)

NAME

tbuffer - streaming tape buffered read

SYNOPSIS

tbuffer [-bn] file

DESCRIPTION

Thuffer allocates n buffers in memory, reads from file into the buffers, and writes from the buffers to the standard output. Each buffer is 61k, and n defaults to 16. When used to buffer a streaming tape, thuffer keeps the tape streaming during the read cycle. This can speed up a restore(8) from a streaming tape, and is used on the installation miniroot diskette.

A maximum of 66 buffers is allowed.

DIAGNOSTICS

Prints messages if it can't allocate the buffers in memory, or if it gets a read or write error.

BUGS

If there is not enough real memory for the buffers, the system will have to swap, and the tape will stop streaming.

VDABORT(8)

NAME

vdabort - abort and spin down a drive

SYNOPSIS

vdabort drive

DESCRIPTION

vdabort will spindown and abort an RVD pack associated with drive which is single digit integer denoting the drive number to be aborted.

SEE ALSO

spinup(8), up(8)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

VDDB(8) VDDB(8)

NAME

vddb - Remote Virtual Disk (RVD) data base manager

SYNOPSIS

$$vddb [-d][-n][database]$$

DESCRIPTION

Vddb is the database management tool used to maintain RVD layout and allocation on the physical media used by the server. Options are:

- Turn on debugging. A trace of the control packets sent and received is displayed.
- No connection to server. This is used to modify the data base locally, so that changes are — n deferred until the server is next started.

database

Specifies an alternate file to use instead of /etc/rvd/rvddb. The tail component (what is to the right of the last "/") of the path database will be taken as the server name. An example would be /etc/rvd/db/jinx. This file should be identical to the copy on the target server (here jinx). To ensure this, copy the file remotely (see rcp(1)) to the server host it defines after vddb exits.

COMMANDS

When executed, vddb enters a command loop. The available commands are:

add physical

add and configure a new physical disk

add virtual

same, but a virtual disk

delete virtual

delete a virtual disk

exchange virtual swap the names of two virtual disks help [command]

show help on command

list

list the contents of database

list physical

same, but specific to a physical disk

list virtual

same, but specific to a virtual disk

modify virtual

modify an existing virtual disk

quit

exit the database program

Parts of commands may be abbreviated.

Each operation will prompt for required information. The operations will be confirmed with "Are you sure (y or n)?" before the action is executed.

Add physical requires two parameters: the raw character device to use, and the number of 512 byte blocks contained in the device. The number of blocks can usually be found in /etc/disktab (see disktab(5)). The raw device should be specified in full, e.g. /dev/hd/c for the c partition of hd1.

Add virtual requires 10 parameters: the virtual pack name (a string), a pack unique id (a number), the owner (a brief description of the pack), the passwords for read-only, exclusive and shared mode, the size in 512-byte blocks, the allowable modes, which is a logical OR of 1 (read-only), 2 (shared), and 4 (exclusive), the owning host name (the owning host may spin up the disk without ever giving a password), and the desired physical disk name (see "Add physical"). The response will indicate if the addition was successful.

Delete virtual prompts for one parameter: the name of the virtual disk name to delete. The pack must be spun down.

Modify virtual prompts for the fields filled in at the time the virtual disk was created; all except the password fields default to their current values.

Quit will exit the program.

VDDB(8)

Help will describe the syntax and semantics of a command, or list all commands if none is specified.

DIAGNOSTICS

Unauthorized operation generally indicates an incorrect password. Other messages are intended to be self-explanatory.

FILES

/etc/rvd/rvddb

data base of current disk layout

SEE ALSO

disktab(5), rvddb(5), rvdsrv(8)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

VDSTATS(8)

NAME

vdstats - list client Remote Virtual Disk (RVD) statistics

SYNOPSIS

vdstats
$$[[-a][-q][-s]]$$

DESCRIPTION

Vdstats lists statistics about current virtual disk connections of this client. With no options, it shows active drive numbers, their status, state, size and server. State is the state of the server when last accessed by this client; it might be out of date if there has been no recent activity with the server. Size prints the capacity of the drive in 512 byte blocks. Server prints the name of the server on which the pack resides.

Vdstats accepts the following options:

- -s Prints operating statistics.
- -q Prints queue statistics.
- -a Prints all of the above.

SEE ALSO

spinup(8), spindown(8), rvdsrv(8)

"The Remote Virtual Disk System" in Volume II, Supplementary Documents

WIDTH3812(8) WIDTH3812(8)

NAME

width3812 - build width tables for IBM 3812 Pageprinter fonts

SYNOPSIS

```
width3812 [ -S ] [ -c codepage ] [ -s size ... 0 ] [ -n width-file ] fontname
```

DESCRIPTION

Width38/2 reads the IBM 3812 Pageprinter font files and generates a width-file and a set of index tables for the font family specified by fontname. The width-file contains the width of each character in the codepage for each size in the sizes list.

The index tables contain an entry for each character in the *codepage*. There is one table for each size in the *sizes* list. The index table is an index into the raster data for the fonts. The index tables are written in the files *fontname.n.codepage*, where n is the point size. The format of this file is described in *font3812(5)*.

For example, if the fontname is ss.B for Sonoran Sans Scrif Bold, and the codepage is stdcp, then this program would be invoked by:

width3812 -c stdcp -n B ss.B

The width-file will be named B and the index table will be named ss.B.n.stdcp. The raster data and character index files for each size of the font will be read to construct the width-file. In this example, the following files will be read: ss.B.6.dat, ss.B.6.ndx, ss.B.7.dat, ss.B.7.ndx, ss.B.8.dat, ss.B.8.ndx, through ss.B.36.dat, ss.B.36.ndx.

Options:

-S This is a *special* font and should be identified as such in the output file. A special font contains the *troff*(1) special characters.

-c codepage

The codepage file maps the troff character names to the IBM character names. The codepage table can be modified to rename the characters or select additional IBM characters for local use. The IBM character names are documented in the IBM 3800 Font Catalog. A set of three codepage tables have been provided. Stdcp names the ASCII characters and the corresponding IBM character names. Picp names the special characters in the IBM Pi fonts. Fixedcp names the characters in the IBM uniformly-spaced fonts. The default codepage is stdcp.

-s size ...0

A 0-terminated list of the sizes of the font to be included in the width-file. If this option is omitted the program will build a width table for sizes 6, 7, 8, 9, 10, 11, 12, 14, 16, 18, 20, 24, 30 and 36. The raster data and character index files for each size will be read. If the files for a specific size do not exist, that size will be omitted from the width-file.

- n width-file

The name of the width-file. The default is fontname.

FILES

/usr/lib/font/dev3812/fonts/stdcp /usr/lib/font/dev3812/fonts/picp /usr/lib/font/dev3812/fonts/fixedcp /usr/lib/font/dev3812/fonts/*.dat /usr/lib/font/dev3812/fonts/*.ndx codepage for ASCII fonts codepage for Pi and Special fonts codepage for uniformly-spaced fonts 3812 font raster data 3812 font index by IBM character name

SEE ALSO

font3812(5), cvt3812(8)
IBM 3800 Printing Subsystem Model III Font Catalog, SH35-0053

PERMUTED INDEX

	@: arithmetic on shell variables	csh(1)
xclock - X Window System, analog	/ digital clock.	
imp:	1822 network interface	
ddn: DDN Standard Mode X.	25 IMP interface	ddn(4)
lib2648: subroutines for the HP ibmemul: IBM	2648 graphics terminal	lib2648(3X) ibmemul(4)
mset: retrieve ASCII to IBM	3270 keyboard map	mset(1)
mset: retrieve ASCII to IBM	3270 keyboard map	mset(1)
database for mapping ascii keystrokes into IBM	3270 keys. map3270:	map3270(5)
data base for mapping ASCII keystrokes into IBM /cvt20to12, cvt00to12: convert IBM 3820 and IBM	3270 keys. map3270:	map3270(5) cvt3812(8)
font3812: font structures for	3812 fonts	font3812(5)
IBM 3820 and IBM 3800 fonts for use with the IBM	3812 Pageprinter. /cvt20to12, cvt00to12: convert	cvt3812(8)
pprint: print text files on IBM	3812 Pageprinter.	pprint(1)
ppt: spooling system filter for the IBM ptroff: print troff files on IBM	3812 Pageprinter	ppt(8) ptroff(1)
vgrind: grind nice listings of programs for the IBM	3812 Pageprinter	vgrind(1)
width3812: build width tables for IBM	3812 Pageprinter fonts	width3812(8)
ibm3812pp: IBM	3812 Pageprinter server.	ibm3812pp(8)
printer3812: IBM cvt3812, cvt20to12, cvt00to12: convert IBM	3812 Pageprinter status information	printer3812(5) cvt3812(8)
ec:	3Com 10 Mb/s Ethernet interface	ec(4)
mtio: 4.	3/RT magtape interface	mtio(4)
diff3:	3-way differential file comparison.	diff3(1)
openpl et al.: f77 library interface to plot	(3X) libraries plot:	plot(3F) colpro(1)
colpro: column filter for IBM ibmbit, ibmgra, ibmpro: output filters for the IBM	4201 Proprinter	lpfilter(8r)
proff: nroff for the IBM	4201 Proprinter and IBM 5152 Graphics Printer	proff(1)
post-processing filter. prfl: IBM	4201 Proprinter/IBM 5152 Graphics Printer nroff	prfl(1)
mtio: sendbug: mail a system bug report to	4.3/RT magtape interface	mtio(4) sendbug(1)
ibm5081, mpel – IBM	5081 Mega Pel Display interface.	ibm5081(4)
ibm5151, mono: IBM	5151 Monochrome Display interface	ibm5151(4)
output filters for the IBM 4201 Proprinter and IBM	5152 Graphics Printer. ibmbit, ibmgra, ibmpro:	lpfilter(8r)
proff: nroff for the IBM 4201 Proprinter and IBM prfl: IBM 4201 Proprinter/IBM	5152 Graphics Printer	proff(1) prfl(1)
ibm5154, ega: IBM	5154 Enhanced Graphics Display interface	ibm5154(4)
interface. ibm6153, apa8: IBM	6153 Advanced Monochrome Graphics Display	ibm6153(4)
ibm6154, apa8c: IBM	6154 Advanced Color Graphics Display interface	ibm6154(4)
interface. ibm6155, apa16: IBM	6155 Extended Monochrome Graphics Display 8514/A Display adapter for the ibm8503, ibm8513,	ibm6155(4)
ibm8514, ibm8604. ibm8514: IBM crl: VAX	8600 console RL02 interface	ibm8514(4) crl(4)
scsiformat: format the IBM	9332 disk unit	scsiformat(8c)
Interface (SCSI) Adapter. sc: IBM	9332 disks using the IBM Small Computer System	sc(4)
vdabort:	abort and spin down a drive	vdabort(8)
	abort: generate a fault	abort(3) abort(3F)
abort: terminate	abruptly with memory image	
	abs: integer absolute value	abs(3)
abs: integer	absolute value.	abs(3)
hypot, cabs: Euclidean distance, complex fabs, floor, ceil:	absolute value	hypot(3M) floor(3M)
	ac: login accounting.	ac(8)
aedemul: graphics interfaces for the IBM	Academic Information Systems experimental display	aedemul(4)
interface. ibmaed, aed: IBM self-tests, aedtest: IBM	Academic Information Systems experimental display Academic Information Systems experimental display .	ibmaed(4) aedtest(8)
sen-tests. acutest. Inwi	acc: ACC LH/DH IMP interface	acc(4)
hdh:	ACC IF-11/HDH IMP interface	hdh(4)
acc:	ACC LH/DH IMP interface	acc(4)
test, and bring online the Advanced Floating Point fpa: direct interface to floating point	Accelerator. afpacode: load,	afpacode(8r) fpa(3X)
accept:	accept a connection on a socket.	accept(2)
•	accept: accept a connection on a socket	accept(2)
setscreen: control display screen	access.	setscreen(8)
xhost - X window system	access: determine accessibility of a file.	xhost(1) access(3F)
	access: determine accessibility of file	access(2)
getgroups: get group	access list.	getgroups(2)
setgroups: set group	access list.	setgroups(2)
initgroups: initialize group bus: control of	access list	initgroups(3) bus(4)
access: determine	accessibility of a file.	access(3F)
		` ',

access: determine	accessibility of file.	access(2)
ac: login	accounting.	ac(8)
sa, accton: system	accounting.	sa(8)
acct: execution	accounting file.	acct(5)
pac: printer/plotter acct: turn	accounting information	pac(8) acct(2)
acct. turn	acct: execution accounting file.	
	acct: turn accounting on or off.	acct(2)
sa,	accton: system accounting.	sa(8)
their inverses. sin, cos, tan, asin,	acos, atan, atan2: trigonometric functions and	sin(3M)
asinh,	acosh, atanh: inverse hyperbolic functions	asinh(3M)
statistics. vdstats:	acquire client Remote Virtual Disk (RVD)	vdstats(2)
signal: change the	action for a signal	signal(3F)
ad: Data Translation	A/D converter.	ad(4)
lant IPM DT DC Token Ding	ad: Data Translation A/D converter	ad(4)
lan: IBM RT PC Token-Ring landump: dump IBM Token-Ring Personal Computer		lan(4)
the IBM Small Computer System Interface (SCSI)	Adapter. sc: IBM 9332 disks using	landump(8r) sc(4)
ibm8514: IBM 8514/A Display	adapter for the ibm8503, ibm8513, ibm8514, ibm8604.	
un: IBM RT PC Baseband	Adapter for use with Ethernet	un(4)
	adb: debugger.	adb(1)
	adb: debugger.	adb(1)
swapon:	add a swap device for interleaved paging/swapping	
·	addbib: create or extend bibliographic database	addbib(1)
adduser: procedure for	adding new users	adduser(8)
swapon: specify	additional device for paging and swapping	swapon(8)
ns_addr, ns_ntoa: Xerox NS(tm)	address conversion routines	
inet_makeaddr, inet_lnaof, inet_netof: Internet	address manipulation routines. /inet_ntoa,	inet(3N)
loc: return the	address of an object.	loc(3F)
getfpemulator: return	address of the floating-point emulator address resolution display and control	getfpemulator(2)
arp: arp:	Address Resolution Protocol	arp(8C) arp(4P)
arp.	adduser: procedure for adding new users	adduser(8)
of the system clock.	adjtime: correct the time to allow synchronization	adjtime(2)
ibm6154, apa8c: IBM 6154	Advanced Color Graphics Display interface	ibm6154(4)
aspacode: load, test, and bring online the	Advanced Floating Point Accelerator	afpacode(8r)
ibm6153, apa8: IBM 6153	Advanced Monochrome Graphics Display interface	
flock: apply or remove an	advisory lock on an open file	flock(2)
dumpaed: dump	aed display memory as a binary file	
display interface. ibmaed,	aed: IBM Academic Information Systems experimental	ibmaed(4)
Information Systems experimental display.	aedemul: graphics interfaces for the IBM Academic	aedemul(4)
	aedjournal: display commands in a log file	aedjournal(1)
experimental display self-tests.	aedrunner: execute graphics commands in a log file aedtest: IBM Academic Information Systems	aedrunner(1) aedtest(8)
yes: be repetitively	affirmative	yes(1)
basename: strip filename	affixes.	basename(1)
Floating Point Accelerator.	afpacode: load, test, and bring online the Advanced	
mem, kmem, kmem1, kmem2, kmem4, ros,	aspamem: main memory	mem(4)
learn: computer	aided instruction about UNIX	learn(1)
learn: computer	aided instruction about UNIX	learn(1)
	alarm: execute a subroutine after a specified time	alarm(3F)
t ti tition i	alarm: schedule signal after specified time	alarm(3C)
L.aliases: UUCP hostname	alias file.	Laliases(5)
unalias: remove	alias: shell macros	csh(1) csh(1)
diffalias. Telliove	aliases: aliases file for sendmail.	aliases(5)
which: locate a program file including	aliases and paths (csh only)	which(1)
newaliases: rebuild the data base for the mail	aliases file.	newaliases(1)
aliases:	aliases file for sendmail	aliases(5)
L.	aliases: UUCP hostname alias file	L.aliases(5)
valloc:	aligned memory allocator.	valloc(3C)
malloc, free, realloc, calloc,	alloca: memory allocator	malloc(3)
malloc, free, realloc, calloc, alloca: memory	allocator	malloc(3)
malloc, free, falloc: memory	allocator	malloc(3F)
valloc: aligned memory scandir,	allocator	valloc(3C) scandir(3)
limit:	alter per-process resource limitations.	csh(1)
renice:	alter priority of running processes	renice(8)
else:	alternative commands	csh(1)
xclock - X Window System,	analog / digital clock	
lex: generator of lexical	analysis programs	lex(1)
error:	analyze and disperse compiler error messages	error(1)
error:	analyze and disperse compiler error messages	error(1)
style:	analyze surface characteristics of a document	
discipline.	ap: asynchronous data mode protocol line	
dumpapa16: dump	apa16 display memory as a binary file	dumpapa16(1)

Display interface. ibm6155,	apa16: IBM 6155 Extended Monochrome Graphics	ibm6155(4)
dumpapa8: dump	apa8 display memory as a binary file	dumpapa8(1)
interface. ibm6153,	apa8: IBM 6153 Advanced Monochrome Graphics Display	ibm6153(4)
dumpapa8c: dump	apa8c display memory as a binary file	dumpapa8c(1)
interface. ibm6154,	apa8c: IBM 6154 Advanced Color Graphics Display .	ibm6154(4)
sendapar: send	APAR	sendapar(8)
sticky: persistent text and	append-only directories.	sticky(8)
.PP uwm - Window Manager Client	Application of X .PP	uwm(1)
apply:	apply a command to a set of arguments	apply(1)
-r.v.	apply: apply a command to a set of arguments	apply(1)
flock:	apply or remove an advisory lock on an open file	flock(2)
nock.	apropos: locate commands by keyword lookup	
		apropos(1)
	ar: archive and library maintainer	ar(1)
	ar: archive (library) file format.	ar(5)
bc:	arbitrary-precision arithmetic language	bc(1)
graphics/ plot: openpl, erase, label, line, circle,	arc, move, cont, point, linemod, space, closepl:	plot(3X)
tp: manipulate tape	archive	tp(1)
ar:	archive and library maintainer	ar(1)
tar: tape	archive file format	tar(5)
ar:	archive (library) file format	ar(5)
tar: tape	archiver	tar(1)
tar: tape	archiver	tar(1)
arff, flcopy:	archiver and copier for floppy	arfi(8V)
ranlib: convert	archives to random libraries.	ranlib(1)
VI_Copy: copy an	area. of the hard disk	copy(3G)
disk: format of reserved	areas of the hard disk	disk(4)
1.1.61	arff, flcopy: archiver and copier for floppy	arff(8V)
glob: filename expand	argument list.	csh(1)
shift: manipulate	argument list.	csh(1)
varargs: variable	argument list.	varargs(3)
apply: apply a command to a set of	arguments	apply(1)
echo: echo	arguments	csh(1)
echo: echo	arguments.	echo(1)
getarg, jarge: return command line	arguments	getarg(3F)
expr: evaluate	arguments as an expression.	expr(1)
getopt: get option letter from	argv	getopt(3)
m out, sdiv, itom: multiple precision integer	arithmetic. /omin, fmin, m in, mout, omout, fmout,	
	arithmetic errors.	mp(3X)
traper: trap		traper(3F)
bc: arbitrary-precision	arithmetic language	bc(1)
@:	arithmetic on shell variables.	csh(1)
	arp: address resolution display and control	arp(8C)
	arp: Address Resolution Protocol	arp(4P)
biff: be notified if mail	arrives and who it is from.	biff(1)
dumpaed: dump aed display memory	as a binary file	dumpaed(1)
dumpapa16: dump apa16 display memory	as a binary file	dumpapa16(1)
dumpapa8: dump apa8 display memory	as a binary file	dumpapa8(1)
dumpapa8c: dump apa8c display memory	as a binary file	dumpapa8c(1)
expr: evaluate arguments	as an expression.	expr(1)
onpir ormanic in garrionic	as: assembler.	as(1)
slattach: attach serial lines		slattach(8C)
Stattacht, attach serial filles	as: VAX-11 assembler.	as(1)
gmtime, asctime, timezone: convert date and time to	ASCII. ctime, localtime,	ctime(3)
	· · · · · · · · · · · · · · · · · · ·	• •
od: octal, decimal, hex,	ascii dump.	od(1)
map3270: database for mapping	ascii keystrokes into IBM 3270 keys	map3270(5)
map3270: data base for mapping	ASCII keystrokes into IBM 3270 keys	map3270(5)
fdate: return date and time in an	ASCII string.	fdate(3F)
mset: retrieve	ASCII to IBM 3270 keyboard map	mset(1)
mset: retrieve	ASCII to IBM 3270 keyboard map	mset(1)
atof, atoi, atol: convert	ASCII to numbers.	atof(3)
ctime, localtime, gmtime,	asctime, timezone: convert date and time to ASCII	ctime(3)
and their inverses. sin, cos, tan,	asin, acos, atan, atan2: trigonometric functions	sin(3M)
. ,	asinh, acosh, atanh: inverse hyperbolic functions	asinh(3M)
as: VAX-11	assembler	as(1)
as:	assembler	as(1)
a.out:	assembler and link editor output.	a.out(5)
a.out:	assembler and link editor output.	a.out(5)
a.out.	assert: program verification	assert(3)
eather eatherfan eathertf.		
setbuf, setbuffer, setlinebuf:	assign buffering to a stream.	setbuf(3S)
tailor: work station customizing	assistance.	tailor(8)
interface.	asy: multi-port asynchronous communications RS232C .	asy(4)
asy: multi-port	asynchronous communications RS232C interface	asy(4)
ap:	asynchronous data mode protocol line discipline	ap(4)
atrm: remove jobs spooled by	at	atrm(1)
shutdown: close down the system	at a given time.	shutdown(8)
at: execute commands	at a later time.	at(1)
	at: execute commands at a later time	at(1)

nice, nohup: run a command	at low priority (sh only).	
inverses. sin, cos, tan, asin, acos, sin, cos, tan, asin, acos, atan,	atan, atan2: trigonometric functions and their atan2: trigonometric functions and their inverses	
asinh, acosh,	atanh: inverse hyperbolic functions	
, ,	atof, atoi, atol: convert ASCII to numbers	
atof,	atoi, atol: convert ASCII to numbers	-3 /
atof, atoi, interrupt. sigpause:	atol: convert ASCII to numbers	. ` ,
interrupt. signause.	atq: print the queue of jobs waiting to be run.	St (-)
	atrm: remove jobs spooled by at.	** :
slattach:	attach serial lines as network interfaces	slattach(8C)
dn: DN-11	autocall unit interface	` '
code. code.	autoconf: diagnostics from the autoconfiguration autoconf: diagnostics from the autoconfiguration	
autoconf: diagnostics from the	autoconfiguration code	-2.5
autoconf: diagnostics from the	autoconfiguration code	
bugfiler: file bug reports in folders	automatically.	bugfiler(8)
rc: command script for	auto-reboot and daemons.	
xload - X window system load wait:	average display.	
wait.	await completion of process	. 1.1
bg: place job in	background.	
wait: wait for	background processes to complete	
bad144: read/write dec standard 144	bad sector information	
badsect: create files to contain badsect: create files to contain	bad sectors	
information.	bad144: read/write dec standard 144 bad sector	
	badsect: create files to contain bad sectors	badsect(8)
	badsect: create files to contain bad sectors	badsect(8)
gettytab: terminal configuration data	base.	gettytab(5)
hosts: host name data networks: network name data	base	hosts(5) networks(5)
phones: remote host phone number data	base.	phones(5)
printcap: printer capability data	base	printcap(5)
protocols: protocol name data	base	protocols(5)
services: service name data	base	
termcap: terminal capability data vgrindefs: vgrind's language definition data	base	
keys. map3270: data	base for mapping ASCII keystrokes into IBM 3270 .	map3270(5)
newaliases: rebuild the data	base for the mail aliases file	
vddb: Remote Virtual Disk (RVD) data	base manager.	
fetch, store, delete, firstkey, nextkey: data dbm_nextkey, dbm_error, dbm_clearerr: data	base subroutines. dbminit,	
un: IBM RT PC	Baseband Adapter for use with Ethernet	
vi: screen oriented (visual) display editor	based on ex.	` '
xcalc: X	based scientific calculator	
	basename: strip filename affixes	
bcopy,	bc: arbitrary-precision arithmetic language bcmp, bzero, ffs: bit and byte string operations	· ,
operations.	bcopy, bcmp, bzero, ffs: bit and byte string	
cb: C program	beautifier	cb(1)
file. VI_Login, VI_Logout:	begin logging subroutine calls and close a log	• • • • • • • • • • • • • • • • • • • •
va: vfont: font formats for the	Benson-Varian interface	` '
j0, j1, jn, y0, y1, yn:	Benson-Varian or Versatec	
J~, Jt, Ju, J°, J1, J1.	bessel functions: of two kinds for integer orders	
random, drandm, irandm:	better random number generator	
changing/ random, srandom, initstate, setstate:	better random number generator; routines for	
addbib: create or extend	bibliographic database	
roffbib: run off	bibliographic database	roffbib(1)
sortbib: sort	bibliographic database	sortbib(1)
index for a bibliography, find references in a	bibliography. indxbib, lookbib: build inverted	
indxbib, lookbib: build inverted index for a	bibliography, find references in a bibliography	
from. comsat:	biff: be notified if mail arrives and who it is biff server	
install: install	binaries.	
whereis: locate source,	binary, and or manual for program	whereis(1)
dumpaed: dump aed display memory as a	binary file.	
dumpapa16: dump apa16 display memory as a dumpapa8: dump apa8 display memory as a	binary file	dumpapa16(1) dumpapa8(1)
dumpapa8c: dump apa8c display memory as a	binary file.	
find the printable strings in a object, or other	binary, file. strings:	• • • •
fread, fwrite: buffered	binary input/output.	
bind:	bind a name to a socket	bind(2)
	bind: bind a name to a socket	bind(2)

	binmail: send or receive mail among users	binmail(1)
bcopy, bcmp, bzero, ffs:	bit and byte string operations	bstring(3)
functions.	bit: and, or, xor, not, rshift, Ishift bitwise	bit(3F)
Tunctions.	bitmap: bitmap editor for X window system	bitmap(1)
bitprt: capture the image on a	bitmap display and print it on an IBM printer	bitprt(1)
bitmap:	bitmap editor for X window system	bitmap(1)
scale: resize a	bitmap image	scale(1)
print it on an IBM printer.		
bit: and, or, xor, not, rshift, lshift	bitprt: capture the image on a bitmap display and bitwise functions	bitprt(1)
communication (obsolete).	bk: line discipline for machine-machine	bit(3F)
sync: update the super	block	bk(4)
		sync(8)
update: periodically update the super	block	update(8)
sigblock:	block signals.	sigblock(2)
sigpause: atomically release	blocked signals and wait for interrupt	sigpause(2)
sum: sum and count	blocks in a file.	sum(1)
fdisk:	boot record partition table maintenance utility	fdisk(8)
reboot: UNIX	bootstrapping procedures	reboot(8)
reboot:	bootstrapping procedures.	reboot(8)
switch: multi-way command	branch.	csh(1)
login,/ sh, for, case, if, while, :, .,	break, continue, cd, eval, exec, exit, export,	sh(1)
	break: exit while/foreach loop.	csh(1)
	breaksw: exit from switch	csh(1)
fg:	bring job into foreground.	csh(1)
Accelerator. afpacode: load, test, and	bring online the Advanced Floating Point	afpacode(8r)
	brk, sbrk: change data segment size	brk(2)
	busemul: kernel bussering emulator	bufemul(4)
ik: Ikonas frame	buffer, graphics device interface	ik(4)
fread, fwrite:	buffered binary input/output	fread(3S)
stdio: standard	buffered input/output package.	stdio(3S)
tbuffer: streaming tape	buffered read.	tbuffer(8)
bufemul: kernel	buffering emulator.	bufemul(4)
setbuf, setbuffer, setlinebuf: assign	buffering to a stream.	setbuf(3S)
generate a dump of the operating system's profile	buffers. kgmon:	kgmon(8)
sendbug: mail a system	bug report to 4bsd-bugs.	sendbug(1)
bugfiler: file		
	bug reports in folders automatically	bugfiler(8)
automatically.	bugfiler: file bug reports in folders	bugfiler(8)
references in a bibliography. indxbib, lookbib:	build inverted index for a bibliography, find	lookbib(1)
mknod:	build special file.	mknod(8)
config.	build system configuration files	config(8)
config:		0(1)
config:	build system configuration files	config(8)
config: width3812:		config(8) width3812(8)
config:	build system configuration files	config(8)
config: width3812: bus: control of access to the system I/O	build system configuration files	config(8) width3812(8)
config: width3812:	build system configuration files	config(8) width3812(8) bus(4)
config: width3812: bus: control of access to the system I/O	build system configuration files	config(8) width3812(8) bus(4) bus(4)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network	build system configuration files	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and	build system configuration files	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp,	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib:	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb:	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C compiler. C canguage X Window System Interface Library. C library functions. C program beautifier.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C compiler. C language X Window System Interface Library. C library functions. C program beautifier. C program source. C programs to implement shared strings. C source.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C compiler. C language X Window System Interface Library. C library functions. C program beautifier. C program source. C programs to implement shared strings. C source.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C compiler. C language X Window System Interface Library. C library functions. C program beautifier. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot,	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart:	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart:	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C program to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default he: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart: diskpart: dc: desk	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes. calculate.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cald (1) diskpart(8) diskpart(8) dc(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart: dc: desk xcalc: X based scientific	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes. calculator.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8) dc(1) xcalc(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default he: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart: diskpart: dc: desk	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program source. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes. calculator. calculator. calculator.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mystr(1) hypot(3M) cal(1) diskpart(8) dc(1) xcalc(1) cal(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: dc: desk xcalc: X based scientific cal: print	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes. calculator. calculator. calendar. calendar. calendar: reminder service.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mystr(1) hypot(3M) cal(1) diskpart(8) dc(1) xcalc(1) cal(1) calendar(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: dc: desk xcalc: X based scientific cal: print syscall: indirect system	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculator. calculator. calculator. calculator. calendar. calendar: reminder service. call.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8) dc(1) xcalc(1) cal(1) calendar(1) syscall(2)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart: diskpart: dc: desk xcalc: X based scientific cal: print syscall: indirect system gprof: display	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes. calculator. calculator. calendar.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8) dc(1) xcalc(1) cal(1) calendar(1) syscall(2) gprof(1)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart: diskpart: dc: desk xcalc: X based scientific cal: print syscall: indirect system gprof: display syscall: system	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes. calculator. calculator. calendar: reminder service. call. call graph profile data. call interface program.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8) diskpart(8) dc(1) xcalc(1) calendar(1) syscall(2) gprof(1) syscall(8)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default he: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart: diskpart: dc: desk xcalc: X based scientific cal: print syscall: indirect system gprof: display syscall: system getuid, getgid: get user or group ID of the	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes. calculator. calculator. calculator. calendar: reminder service. call. call graph profile data. call interface program. caller.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8) diskpart(8) dc(1) xcalc(1) cal(1) calendar(1) syscall(2) gprof(1) syscall(8) getuid(3F)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default he: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart: diskpart: dc: desk xcalc: X based scientific cal: print syscall: indirect system gprof: display syscall: system getuid, getgid: get user or group ID of the malloc, free, realloc,	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes. calculator. calculator. calculator. calendar: reminder service. call. call graph profile data. call interface program. caller. calloc, alloca: memory allocator.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8) diskpart(8) dc(1) xcalc(1) cal(1) calendar(1) syscall(2) gprof(1) syscall(8) getuid(3F) malloc(3)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default he: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: dc: desk xcalc: X based scientific cal: print syscall: indirect system gprof: display syscall: system getuid, getgid: get user or group ID of the malloc, free, realloc, siginterrupt: allow signals to interrupt system	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes. calculator. calculator. calculator. calendar: reminder service. call. call graph profile data. call interface program. caller. calloc, alloca: memory allocator. calls.	config(8) width3812(8) bus(4) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8) dc(1) xcalc(1) calendar(1) syscall(2) gprof(1) syscall(8) getuid(3F) malloc(3) siginterrupt(3)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart: dc: desk xcalc: X based scientific cal: print syscall: indirect system gprof: display syscall: system getuid, getgid: get user or group ID of the malloc, free, realloc, siginterrupt: allow signals to interrupt system VI_Login, VI_Logout: begin logging subroutine	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes. calculator. calculator. calculator. calendar: reminder service. call. call graph profile data. call interface program. caller. calloc, alloca: memory allocator. calls. calls and close a log file.	config(8) width3812(8) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8) diskpart(8) dc(1) xcalc(1) cal(1) calendar(1) syscall(2) gprof(1) syscall(8) getuid(3F) malloc(3)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart: dc: desk xcalc: X based scientific cal: print syscall: indirect system gprof: display syscall: system getuid, getgid: get user or group ID of the malloc, free, realloc, siginterrupt: allow signals to interrupt system VI_Login, VI_Logout: begin logging subroutine intro: introduction to system	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculated calculator. calculator. calculator. calculator. calendar. calendar: reminder service. call. call graph profile data. call interface program. caller. calloc, alloca: memory allocator. calls and close a log file. calls and error numbers.	config(8) width3812(8) bus(4) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8) dc(1) xcalc(1) calendar(1) syscall(2) gprof(1) syscall(8) getuid(3F) malloc(3) siginterrupt(3)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart: dc: desk xcalc: X based scientific cal: print syscall: indirect system gprof: display syscall: system getuid, getgid: get user or group ID of the malloc, free, realloc, siginterrupt: allow signals to interrupt system VI_Login, VI_Logout: begin logging subroutine	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculate default disk partition sizes. calculator. calculator. calculator. calendar: reminder service. call. call graph profile data. call interface program. caller. calloc, alloca: memory allocator. calls. calls and close a log file.	config(8) width3812(8) bus(4) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mkstr(1) hypot(3M) cal(1) diskpart(8) dc(1) xcalc(1) calendar(1) syscall(2) gprof(1) syscall(8) getuid(3F) malloc(3) siginterrupt(3) log(3G)
config: width3812: bus: control of access to the system I/O ntohs: convert values between host and network bcopy, bcmp, bzero, ffs: bit and swab: swap bcopy, bcmp, cc: cc: default hc: High pcc: pcc-based Xlib: intro: introduction to cb: indent: indent and format lint: a xstr: extract strings from mkstr: create an error message file by massaging hypot, diskpart: diskpart: dc: desk xcalc: X based scientific cal: print syscall: indirect system gprof: display syscall: system getuid, getgid: get user or group ID of the malloc, free, realloc, siginterrupt: allow signals to interrupt system VI_Login, VI_Logout: begin logging subroutine intro: introduction to system	build system configuration files. build width tables for IBM 3812 Pageprinter fonts. bus. bus: control of access to the system I/O bus. byte order. htonl, htons, ntohl, byte string operations. bytes. bzero, ffs: bit and byte string operations. C compiler. C compiler. C compiler. C compiler. C Language X Window System Interface Library. C library functions. C program beautifier. C program source. C program verifier. C programs to implement shared strings. C source. cabs: Euclidean distance, complex absolute value. cal: print calendar. calculate default disk partition sizes. calculated calculator. calculator. calculator. calculator. calendar. calendar: reminder service. call. call graph profile data. call interface program. caller. calloc, alloca: memory allocator. calls and close a log file. calls and error numbers.	config(8) width3812(8) bus(4) bus(4) bus(4) byteorder(3N) bstring(3) swab(3) bstring(3) cc(1) cc(1) hc(1) pcc(1) Xlib(3X) intro(3) cb(1) indent(1) lint(1) xstr(1) mystr(1) mystr(1) diskpart(8) dc(1) xcalc(1) cal(1) calendar(1) syscall(2) gprof(1) syscall(8) getuid(3F) malloc(3) siginterrupt(3) log(3G) intro(2)

termcap: terminal	capability data base	termcap(5)
on an IBM printer. bitprt:	capture the image on a bitmap display and print it	bitprt(1)
XMenu - X Deck of	cards Menu System	XMenu(3X)
cd, eval, exec, exit, export, login,/ sh, for,	case, if, while, :, ., break, continue,	sh(1)
, ,	case: selector in switch	csh(1)
tu: VAX-11/730 and VAX-11/750 TU58 console	cassette interface.	tu(4)
uu: TU58/DECtape II UNIBUS	cassette interface.	uu(4)
	cat: catenate and print.	cat(1)
catman: create the	cat files for the manual	
default:	catchall clause in switch.	csh(1)
cat:	catenate and print	• • •
statistics. rvdlog:	catman: create the cat files for the manual cause Remote Virtual Disk (RVD) server to log	
statistics. Tydiog.	cb: C program beautifier	rvdlog(8) cb(1)
	cbrt, sqrt: cube root, square root.	sqrt(3M)
	cc: C compiler.	cc(1)
	cc: default C compiler.	cc(1)
	cd: change directory.	
	cd: change working directory.	
case, if, while, :, ., break, continue,	cd, eval, exec, exit, export, login, read,//for,	
fabs, floor,	ceil: absolute value, floor, ceiling functions	floor(3M)
fabs, floor, ceil: absolute value, floor,	ceiling functions.	floor(3M)
chdir:	change current working directory	chdir(2)
brk, sbrk:	change data segment size	brk(2)
chdir:	change default directory.	chdir(3F)
cd:	change directory.	csh(1)
chdir:	change directory.	csh(1)
ioinit:	change f77 I/O initialization.	ioinit(3F)
chgrp: server. rvdchlog:	change group	
chmod:	change mode	rvdchlog(8) chmod(1)
chmod:	change mode of a file.	chmod(3F)
chmod, fchmod:	change mode of file.	chmod(2)
umask:	change or display file creation mask	csh(1)
chown:	change owner	chown(8)
chown, fchown:	change owner and group of a file	chown(2)
chfn, chsh, passwd:	change password file information	passwd(1)
chroot:	change root directory	chroot(2)
VI_Color:	change screen color	color(3G)
signal:	change the action for a signal	signal(3F)
rename:	change the name of a file.	rename(2)
set:	change value of shell variable	csh(1)
cd:	change working directory.	cd(1)
better random number generator; routines for pipe: create an interprocess communication	changing generators. /srandom, initstate, setstate:	
• •	channel	
ungetc: push iscntrl, isascii, toupper, tolower, toascii:	character classification macros. /isprint, isgraph,	ungetc(3S) ctype(3)
getc, fgetc: get a	character from a logical unit.	getc(3F)
index, rindex, Inblnk, len: tell about	character objects.	index(3F)
getc, getchar, fgetc, getw: get	character or word from stream.	getc(3S)
putc, putchar, fputc, putw: put	character or word on a stream.	putc(3S)
putc, fputc: write a	character to a fortran logical unit	putc(3F)
style: analyze surface	characteristics of a document	style(1)
tr: translate	characters	tr(1)
	chdir: change current working directory	chdir(2)
	chdir: change default directory.	chdir(3F)
dehaals Gle augt-m-din-stans ann-i-t	chdir: change directory.	csh(1)
dcheck: file system directory consistency	check	dcheck(8)
icheck: file system storage consistency fsck: file system consistency	check	icheck(8) fsck(8)
checknr:	check nroff/troff files.	checknr(1)
eqn, neqn,	checkeq: typeset mathematics	eqn(1)
quotacheck: file system quota consistency	checker	quotacheck(8)
fastboot, fasthalt: reboot/halt the system without	checking the disks.	fastboot(8)
	checknr: check nroff/troff files	checknr(1)
information.	chin, chsh, passwd: change password file	passwd(1)
	chgrp: change group.	chgrp(1)
•	chmod: change mode.	chmod(1)
	chmod: change mode of a file	chmod(3F)
	chmod, fchmod: change mode of file	chmod(2)
	chown change owner	chown(8)
	chown, fchown: change owner and group of a file chroot: change root directory	chown(2) chroot(2)
chín,	chsh, passwd: change password file information	passwd(1)
VI Circle: draw a	circle.	circle(3G)
closepl:/ plot: openpl, erase, label, line,	circle, arc, move, cont, point, linemod, space,	plot(3X)
	and the second s	

infinity,/ copysign, drem, logb, scalb, rint,		. (6)
	classdouble, classfloat, isnan, unordered, finite,	ieee(3)
copysign, drem, logb, scalb, rint, classdouble,	classfloat, isnan, unordered, finite, infinity,/	ieee(3)
isascii, toupper, tolower, toascii: character	classification macros. /isprint, isgraph, iscntrl,	ctype(3)
default: catchall	clause in switch.	csh(1)
		, , ,
uuclean: uucp spool directory		uuclean(8C)
	clear: clear terminal screen.	
clri:	clear i-node.	clri(8)
clear:	clear terminal screen	clear(1)
ferror, feof,	clearerr, fileno: stream status inquiries.	
.PP uwm - Window Manager	Client Application of X.PP	
vdstats: acquire	client Remote Virtual Disk (RVD) statistics	vdstats(2)
vdstats: list	client Remote Virtual Disk (RVD) statistics	vdstats(8)
up, down:	client Remote Virtual Disk (RVD) utilities	up(1)
/etc/rvd/rvdtab: information about	client Remote Virtual Disks (RVDs)	rvdtab(5)
rvdflush: spindown	client's Remote Virtual Disk (RVD) packs	rvdflush(8)
csh: a shell (command interpreter) with	C-like syntax	csh(1)
VI Clip: set	clipping window.	
the time to allow synchronization of the system	clock. adjtime: correct	adjtime(2)
kg: KL-11/DL-11W line	clock	kg(4)
xclock - X Window System, analog / digital	clock	xclock(1)
cron:	clock daemon.	cron(8)
VI_Logout: begin logging subroutine calls and	close a log file. VI_Login,	log(3G)
-	close: delete a descriptor	'
shutdown:	close down the system at a given time	shutdown(8)
fclose, fflush:	close or flush a stream.	fclose(3S)
opendir, readdir, telldir, seekdir, rewinddir,	closedir: directory operations	directory(3)
syslog, openlog,	closelog, setlogmask: control system log	
circle, arc, move, cont, point, linemod, space,	closepl: graphics interface. /erase, label, line,	
oncie, are, move, com, point, infeniou, space,		
	clri: clear i-node	
L.	cmds: UUCP remote command permissions file	L.cmds(5)
	cmp: compare two files	cmp(1)
autoconf: diagnostics from the autoconfiguration	code	autoconf(4)
autoconf: diagnostics from the autoconfiguration	code	autoconf(4)
pi: Pascal interpreter	code translator	pi(1)
•	col: filter reverse line feeds	col(1)
	colort: filter nroff output for CRT previewing	colcrt(1)
1		
log. dmesg:	collect system diagnostic messages to form error	dmesg(8)
VI_Color: change screen	color	color(3G)
ibm6154, apa8c: IBM 6154 Advanced	Color Graphics Display interface	ibm6154(4)
, , ,	colpro: column filter for IBM 4201 Proprinter	
<u>, </u>	colrm: remove columns from a file.	
colpro:	column filter for IBM 4201 Proprinter	
colrm: remove	columns from a file	I/1\
VOILILL TOUTO	Columns nom a me.	coirm(1)
files.	comm: select or reject lines common to two sorted	comm(1)
files. exec: overlay shell with specified	comm: select or reject lines common to two sorted command	comm(1) csh(1)
files. exec: overlay shell with specified time: time	comm: select or reject lines common to two sorted	comm(1)
files. exec: overlay shell with specified	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1)
files. exec: overlay shell with specified time: time test: condition	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1)
files. exec: overlay shell with specified time: time test: condition time: time a	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3)
files. exec: overlay shell with specified time: time test: condition time: time a	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1) csh(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1) csh(1) csh(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a	comm: select or reject lines common to two sorted command. command at low priority (sh only). command branch. command hash table. command hash table. command hashing statistics. command immune to hangups. (command interpreter) with C-like syntax. command is.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait:	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command hashing statistics. command interpreter) with C-like syntax. command is. command is. command language. /cxec, exit, export, login, read,	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) sh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command interpreter) with C-like syntax. command is. command language. /exec, exit, export, login, read, command line arguments.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) sh(1) getarg(3F)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait:	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command hashing statistics. command interpreter) with C-like syntax. command is. command is. command language. /cxec, exit, export, login, read,	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) sh(1) getarg(3F)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote	comm: select or reject lines common to two sorted command	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) ch(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command immune to hangups. (command interpreter) with C-like syntax. command is. command language. /cxcc, exit, export, login, read, command line arguments. command permissions file.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) desh(1) de
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc:	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command immune to hangups. (command interpreter) with C-like syntax. command is. command language. /cxcc, exit, export, login, read, command permissions file. command repeatedly. command script for auto-reboot and daemons.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) getarg(3F) L.cmds(5) csh(1) rc(8)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc: onintr: process interrupts in	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command interpreter) with C-like syntax. command is. command language. /exec, exit, export, login, read, command permissions file. command repeatedly. command repeatedly. command script for auto-reboot and daemons. command scripts.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc:	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command immune to hangups. (command interpreter) with C-like syntax. command is. command language. /cxcc, exit, export, login, read, command permissions file. command repeatedly. command script for auto-reboot and daemons.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) getarg(3F) L.cmds(5) csh(1) rc(8)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc: onintr: process interrupts in	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command interpreter) with C-like syntax. command is. command language. /exec, exit, export, login, read, command permissions file. command repeatedly. command repeatedly. command script for auto-reboot and daemons. command scripts.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc: onintr: process interrupts in apply: apply a goto:	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command interpreter) with C-like syntax. command in arguments. command language. /cxcc, exit, export, login, read, command permissions file. command script for auto-reboot and daemons. command to a set of arguments. command to a set of arguments.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) rch(1) sh(1) getarg(3F) L.cmds(5) csh(1) rc(8) csh(1) apply(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc: onintr: process interrupts in apply: apply a goto: else: alternative	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command interpreter) with C-like syntax. command is. command language. /cxec, exit, export, login, read, command permissions file. command repeatedly. command scripts. command to a set of arguments. command to a set of arguments. command transfer. commands.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) getarg(3F) L.cmds(5) csh(1) rc(8) csh(1) apply(1) csh(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc: onintr: process interrupts in apply: apply a goto: else: alternative intro: introduction to	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command interpreter) with C-like syntax. command interpreter) with C-like syntax. command language. /cxec, exit, export, login, read, command permissions file. command repeatedly. command script for auto-reboot and daemons. command to a set of arguments. command to a set of arguments. commands.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) getarg(3F) L.cmds(5) csh(1) rc(8) csh(1) apply(1) csh(1) csh(1) intro(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc: onintr: process interrupts in apply: apply a goto: else: alternative intro introduction to introduction to system maintenance and operation	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command immune to hangups. (command interpreter) with C-like syntax. command language. /exec, exit, export, login, read, command permissions file. command repeatedly. command script for auto-reboot and daemons. command transfer. commands. commands. commands.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) getarg(3F) L.cmds(5) csh(1) rc(8) csh(1) apply(1) csh(1) intro(1) intro(8)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc: onintr: process interrupts in apply: apply a goto: else: alternative intro: introduction to	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command interpreter) with C-like syntax. command interpreter) with C-like syntax. command language. /cxec, exit, export, login, read, command permissions file. command repeatedly. command script for auto-reboot and daemons. command to a set of arguments. command to a set of arguments. commands.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) getarg(3F) L.cmds(5) csh(1) rc(8) csh(1) apply(1) csh(1) csh(1) intro(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc: onintr: process interrupts in apply: apply a goto: else: alternative intro introduction to introduction to system maintenance and operation	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command immune to hangups. (command interpreter) with C-like syntax. command language. /exec, exit, export, login, read, command permissions file. command repeatedly. command script for auto-reboot and daemons. command transfer. commands. commands. commands.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) dsh(1) dsh(1) getarg(3F) L.cmds(5) csh(1) rc(8) csh(1) apply(1) csh(1) intro(1) intro(8) intro(8)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc: onintr: process interrupts in apply: apply a goto: else: alternative introduction to system maintenance and operation at: execute	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command interpreter) with C-like syntax. command is. command language. /exec, exit, export, login, read, command permissions file. command repeatedly. command script for auto-reboot and daemons. command to a set of arguments. command transfer. commands.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) rcmds(5) csh(1) rc(8) csh(1) apply(1) csh(1) intro(8) intro(8) at(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc: onintr: process interrupts in apply: apply a goto: else: alternative introduction to system maintenance and operation at: execute apropos: locate	comm: select or reject lines common to two sorted command. command st low priority (sh only). command hash table. command hash table. command hashing statistics. command interpreter) with C-like syntax. command is. command language. /cxcc, exit, export, login, read, command permissions file. command repeatedly. command script for auto-reboot and daemons. command transfer. commands. com	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) getarg(3F) L.cmds(5) csh(1) rc(8) csh(1) intro(8) intro(8) at(1) appropos(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute reconsister of the construction to introduction to system maintenance and operation introduction to system maintenance and operation at: execute while: repeat	comm: select or reject lines common to two sorted command. command st low priority (sh only). command branch. command hash table. command hash table. command interpreter) with C-like syntax. command is. command line arguments. command permissions file. command script for auto-reboot and daemons. command to a set of arguments. command to a set of arguments. commands.	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) tosh(1) csh(1) intro(8) csh(1) apropos(1) csh(1) csh(1)
files. exec: overlay shell with specified time: time test: condition time: time a routines for returning a stream to a remote rexec: return stream to a remote system: issue a shell system: execute a UNIX nice, nohup: run a switch: multi-way rehash: recompute unhash: discard hashstat: print nohup: run csh: a shell whatis: describe what a readonly, set, shift, times, trap, umask, wait: getarg, iargc: return L.cmds: UUCP remote repeat: execute rc: onintr: process interrupts in apply: apply a goto: else: alternative introduction to system maintenance and operation at: execute apropos: locate	comm: select or reject lines common to two sorted command. command st low priority (sh only). command hash table. command hash table. command hashing statistics. command interpreter) with C-like syntax. command is. command language. /cxcc, exit, export, login, read, command permissions file. command repeatedly. command script for auto-reboot and daemons. command transfer. commands. com	comm(1) csh(1) csh(1) test(1) time(1) rcmd(3) rexec(3) system(3F) nice(1) csh(1) csh(1) csh(1) csh(1) csh(1) csh(1) getarg(3F) L.cmds(5) csh(1) rc(8) csh(1) intro(8) intro(8) at(1) appropos(1)

source: read	commands from file.	csh(1)
aedjournal: display	commands in a log file.	aedjournal(1)
aedrunner: execute graphics comm: select or reject lines	commands in a log file	
socket: create an endpoint for	communication.	comm(1) socket(2)
pipe: create an interprocess	communication channel.	pipe(2)
bk: line discipline for machine-machine	communication (obsolete).	bk(4)
talkd: remote user	communication server.	talkd(8C)
dmc: DEC DMC-11/DMR-11 point-to-point	communications device	dmc(4)
dh: DH-11/DM-11	communications multiplexer	dh(4)
dhu: DHU-11	communications multiplexer	dhù(4)
dz: DZ-11	communications multiplexer	dz(4)
asy: multi-port asynchronous	communications RS232C interface	asy(4)
users:	compact list of users who are on the system	users(1)
diff: differential file and directory	comparator	diff(1)
cmp:	compare two files.	cmp(1)
diff3: 3-way differential file	comparison.	diff3(1)
liszt:	compile a Franz Lisp program	liszt(1)
cc: C	compiler	cc(1)
cc: default C	compiler	∞(1)
f77: Fortran 77	compiler	f77(1)
f77: FORTRAN 77	compiler	177(1)
hc: High C	compiler	hc(1)
pc: Pascal	compiler	pc(1)
pcc: pcc-based C	compiler	pcc(1)
pp: Professional Pascal	compiler	1117
error: analyze and disperse error: analyze and disperse	compiler error messages.	
yacc: yet another	compiler error messages	error(1) yacc(1)
fp: Functional Programming language	compiler/interpreter	fp(1)
wait: wait for background processes to	complete	csh(1)
wait: wait for background processes to	completion of process.	wait(1)
hypot, cabs: Euclidean distance,	complex absolute value.	hypot(3M)
compress, uncompress, zcat:	compress and expand data	compress(1)
data.	compress, uncompress, zcat: compress and expand	compress(1)
landump: dump IBM Token-Ring Personal	Computer Adapter	landump(8r)
learn:	computer aided instruction about UNIX	learn(1)
learn:	computer aided instruction about UNIX	learn(1)
sc: IBM 9332 disks using the IBM Small	Computer System Interface (SCSI) Adapter	sc(4)
	comsat: biff server	comsat(8C)
test:	condition command	test(1)
endif: terminate	conditional	csh(1)
if:	conditional statement	csh(1)
while: repeat commands	conditionally.	csh(1)
	config: build system configuration files	config(8)
	config: build system configuration files	config(8)
rc.		rc.config(5)
gettytab: terminal	U	
	configuration file.	
	configuration file for startup scripts	rc.config(5)
config: build system	configuration files.	config(8)
config: build system	configuration files	config(8)
rvddb: Remote Virtual Disk (RVD) server	configuration table	rvddb(5)
ifconfig:	configure network interface parameters	ifconfig(8C)
ifconfig:	configure network interface parameters	ifconfig(8c)
getpeername: get name of	connect: initiate a connection on a socket	connect(2) getpeername(2)
socketpair: create a pair of	connected sockets.	socketpair(2)
shutdown: shut down part of a full-duplex	connection.	shutdown(2)
accept: accept a		accept(2)
connect: initiate a	connection on a socket	connect(2)
listen: listen for	connections on a socket.	listen(2)
rydshow: show	connections to Remote Virtual Disk (RVD) server	rvdshow(8)
	cons: keyboard and console display interface	cons(4)
	cons: VAX-11 console interface	cons(4)
dcheck: file system directory	consistency check	dcheck(8)
icheck: file system storage	consistency check	icheck(8)
fsck: file system	consistency check and interactive repair	fsck(8)
quotacheck: file system quota	consistency checker	quotacheck(8)
tu: VAX-11/730 and VAX-11/750 TU58	console cassette interface	tu(4)
cons: keyboard and	console display interface	cons(4)
fl:	console floppy interface	fl(4)
cons: VAX-11	console interface.	cons(4)
crl: VAX 8600	console RI.02 interface	cri(4)
speaker:	console speaker interface	speaker(4)
	consoles: utility database of display screens	consoles(5)

show what versions of object modules were used to	construct a Gla subati	mbat/1)
· · · · · · · · · · · · · · · · · · ·	construct a file. what:	what(1)
mkfs: newfs:	construct a file system.	mkfs(8)
newis:	construct a new file system	newfs(8) newfs(8)
mkproto:	construct a new file system	` '
deroff: remove nroff, troff, tbl and eqn	constructs,	deroff(1)
setrlimit: control maximum system resource	consumption. getrlimit,	
vlimit: control maximum system resource	consumption	vlimit(3C)
openpl, erase, label, line, circle, arc, move,	cont, point, linemod, space, closepl: graphics/	plot(3X)
badsect: create files to	contain bad sectors.	badsect(8)
badsect: create files to	contain bad sectors.	badsect(8)
ls: list		ls(1)
rvdcopy: copy	contents of directory	
	•	
sigstack: set and/or get signal stack	context.	
sh, for, case, if, while, :, ., break,	continue, cd, eval, exec, exit, export, login,/	
	continue: cycle in loop.	`'
arp: address resolution display and	control	- * *
fcntl: file	control	fcntl(2)
ioctl:	control device.	. ,
setscreen:	control display screen access	` '
init: process	control initialization.	init(8)
init: process	control initialization	
getrlimit, setrlimit:	control maximum system resource consumption	getrlimit(2)
vlimit:	control maximum system resource consumption	
icmp: Internet	Control Message Protocol	` ,
bus:	control of access to the system I/O bus	bus(4)
lpc: line printer	control program.	lpc(8)
timedc: timed	control program.	timedc(8)
xhost - X window system access	control program.	
tcp: Internet Transmission	Control Protocol.	
rvdsend - send	control stream to Remote Virtual Disk (RVD) server.	rvdsend(8)
	· · · · · · · · · · · · · · · · · · ·	
syslog, openlog, closelog, setlogmask:	control system log.	
vhangup: virtually "hangup" the current	control terminal.	vhangup(2)
VI_FDefnCur, VI_EnCur, VI_DisCur, VI_PosnCur:	control the display cursor. VI_MDefnCur,	
uda: UDA-50 disk	controller interface	uda(4)
up: unibus storage module	controller/drives	up(4)
ecvt, fcvt, gcvt: output	conversion	ecvt(3)
long, short: integer object	conversion	long(3F)
printf, fprintf, sprintf: formatted output	conversion	printf(3S)
scanf, fscanf, sscanf: formatted input	conversion	scanf(3S)
units:	conversion program	units(1)
ns_addr, ns_ntoa: Xerox NS(tm) address	conversion routines	ns(3N)
dd:	convert and copy a file	
ranlib:	convert archives to random libraries	
atof, atoi, atol:	convert ASCII to numbers	
ctime, localtime, gmtime, asctime, timezone:	convert date and time to ASCII	
the IBM 3812/ cvt3812, cvt20to12, cvt00to12:	convert IBM 3820 and IBM 3800 fonts for use with .	cvt3812(8)
htable:	convert NIC standard format host tables	` '
cvtsym:	convert symbol table.	`
	convert values between host and network byte order	
htonl, htons, ntohl, ntohs:		byteorder(3N)
ad: Data Translation A/D	converter.	ad(4)
flcopy:	copier for diskettes	flcopy(8r)
arff, flcopy: archiver and	copier for floppy.	• •
cp:	copy	
dd: convert and	copy a file.	
tcopy:	copy a mag tape	
VI_Copy:	copy an area.	
rvdcopy:	copy contents of one RVD disk pack to another	1 - 1 /
fork: create a	copy of this process.	
remainder, exponent manipulations.	copysign, drem, finite, logb, scalb: copysign,	ieee(3M)
classfloat, isnan, unordered, finite, infinity,/	copysign, drem, logb, scalb, rint, classdouble,	ieee(3)
copysign, drem, finite, logb, scalb:	copysign, remainder, exponent manipulations	ieee(3M)
nfabort: dump	core and log it in a notesfile	nfabort(3)
savecore: save a	core dump of the operating system	
	core: format of memory image file	
	core: format of memory image file	
gcore: get	core images of running processes.	
system clock. adjtime:	correct the time to allow synchronization of the	
functions and their inverses. sin,	cos, tan, asin, acos, atan, atan2: trigonometric	sin(3M)
sinh,	cosh, tanh: hyperbolic functions	
wc: word	count.	
sum: sum and	count blocks in a file.	
Suin. Sum and		` '
	cp: copy.	
	crash: what happens when the system crashes	
arachi what hamana when thet	crash: what happens when the system crashes	
crash: what happens when the system	crashes	crash(8r)

crash: what happens when the system	crashes.	crash(8V)
	creat: create a new file	creat(2)
fork:	create a copy of this process.	fork(3F)
creat:	create a new file	creat(2)
open: open a file for reading or writing, or (RVD). newvd:	create a new file	open(2) newvd(8)
fork:	create a new process	fork(2)
socketpair:	create a pair of connected sockets.	socketpair(2)
ctags:	create a tags file	ctags(1)
socket:	create an endpoint for communication	socket(2)
mkstr:	create an error message file by massaging C source	mkstr(1)
pipe:	create an interprocess communication channel	pipe(2)
badsect: badsect:	create files to contain bad sectors	badsect(8) badsect(8)
addbib:	create or extend bibliographic database	addbib(1)
catman:	create the cat files for the manual	catman(8)
umask: change or display file	creation mask.	csh(1)
umask: set file	creation mode mask	umask(2)
	crl: VAX 8600 console RI.02 interface	crl(4)
	cron: clock daemon.	cron(8)
lxref: lisp	cross reference program	lxref(1)
pxref: Pascal	cross-reference program.	pxref(1)
systat: display system statistics on a colort: filter nroff output for	CRT previewing.	systat(1) colcrt(1)
more, page: file perusal filter for	crt viewing.	more(1)
more, page. me per usar meer rer	crypt: encode/decode.	crypt(1)
	crypt, setkey, encrypt: DFS encryption	crypt(3)
syntax.	csh: a shell (command interpreter) with C-like	csh(1)
ŕ	css: DEC IMP-11A LH/DH IMP interface	css(4)
pcl: DEC	CSS PCL-11 B Network Interface	pcl(4)
	ct: phototypesetter interface	ct(4)
	ctags: create a tags file	ctags(1)
convert date and time to ASCII.	ctime, localtime, gmtime, asctime, timezone:	ctime(3)
time,	ctime, Itime, gmtime: return system time	time(3F)
cbrt, sqrt:	cube root, square root.	sqrt(3M)
vhangup: virtually "hangup" the	current control terminal	vhangup(2)
gethostid, sethostid: get/set unique identifier of gethostname, sethostname: get/set name of	current host.	gethostid(2) gethostname(2)
hostnm: get name of	current host.	hostnm(3F)
hostid: set or print identifier of	current host system.	hostid(1)
hostname: set or print name of	current host system.	hostname(1)
jobs: print	current job list	csh(1)
VI_AMove, VI_RMove: move the	current point	move(3G)
sigsetmask: set	current signal mask	sigsetmask(2)
whoami: print effective	current user id.	whoami(1)
chdir: change getcwd: get pathname of	current working directory	chdir(2) getcwd(3F)
getwd: get pathname of getwd: get wd: get	current working directory	
motion.	curses: screen functions with "optimal" cursor	curses(3X)
VI DisCur, VI PosnCur: control the display	cursor. VI_MDefnCur, VI_FDefnCur, VI_EnCur,	cursor(3G)
curses: screen functions with "optimal"	cursor motion.	curses(3X)
spline: interpolate smooth	curve	spline(1G)
tailor: work station	customizing assistance	tailor(8)
use with the IBM 3812/ cvt3812, cvt20to12,	cvt00to12: convert IBM 3820 and IBM 3800 fonts for	cvt3812(8)
fonts for use with the IBM 3812/ cvt3812,	cvt20to12, cvt00to12: convert IBM 3820 and IBM 3800	cvt3812(8)
IBM 3800 fonts for use with the IBM 3812/	cvt3812, cvt20to12, cvt00to12: convert IBM 3820 and	cvt3812(8)
continue:	cvtsym: convert symbol table	cvtsym(8) csh(1)
cron: clock	daemon.	cron(8)
lpd: line printer	daemon.	lpd(8)
routed: network routing	daemon	routed(8C)
rvdsrv: Remote Virtual Disk (RVD) server	daemon	rvdsrv(8)
timed: time server	daemon	timed(8)
XNSrouted: NS Routing Information Protocol	daemon.	XNSrouted(8C)
rc: command script for auto-reboot and	daemons.	rc(8)
ftpd:	DARPA Internet File Transfer Protocol server	ftpd(8C)
whois:	DARPA Internet user name directory service	whois(1)
telnetd:	DARPA TELNET protocol server	telnetd(8C) tftpd(8C)
tftpd: VI Dash: set line	DARPA Trivial File Transfer Protocol server dash pattern	dash(3G)
compress, uncompress, zcat: compress and expand	data	compress(1)
eval: re-evaluate shell	data.	csh(1)
gprof: display call graph profile	data.	gprof(1)
prof: display profile	data	prof(1)
ttys: terminal initialization	data	ttys(5)
VI MRead, VI FRead: read display	data	read(3G)

gettytab: terminal configuration		
	data base	gettytab(5)
hosts: host name	data base	hosts(5)
networks: network name	data base	networks(5)
phones: remote host phone number	data base.	phones(5)
printcap: printer capability	data base	printcap(5)
protocols: protocol name	data base.	protocols(5)
services: service name	data base.	services(5)
termcap: terminal capability	data base.	termcap(5)
vgrindefs: vgrind's language definition	data base.	vgrinde(s(5)
3270 keys. map3270:	data base for mapping ASCII keystrokes into IBM	map3270(5)
newaliases: rebuild the	data base for the mail aliases file.	newaliases(1)
vddb: Remote Virtual Disk (RVD)	data base manager.	vddb(8)
dbminit, fetch, store, delete, firstkey, nextkey:	data base subroutines.	dbm(3X)
dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr:	data base subroutines. /dbm store, dbm delete,	ndbm(3)
	data mode protocol line discipline.	
ap: asynchronous	· · · · · · · · · · · · · · · · · · ·	ap(4)
brk, sbrk: change	data segment size.	brk(2)
null:	data sink.	null(4)
ad:	Data Translation A/D converter	ad(4)
types: primitive system	data types.	types(5)
addbib: create or extend bibliographic	database	addbib(1)
roffbib: run off bibliographic	database	roffbib(1)
sortbib: sort bibliographic	database	sortbib(1)
keys. map3270:	database for mapping ascii keystrokes into IBM 3270 .	map3270(5)
consoles: utility	database of display screens	consoles(5)
join: relational	database operator	join(1)
xrdb - Server Resource	Database Utility	xrdb(1)
idp: Xerox Internet	Datagram Protocol.	idp(4P)
udp: Internet User	Datagram Protocol.	udp(4P)
date: print and set the	date.	date(1)
date: print and set the	date.	date(1)
		• •
gettimeofday, settimeofday: get/set	date and time.	gettimeofday(2)
time, ftime: get	date and time.	time(3C)
fdate: return	date and time in an ASCII string.	fdate(3F)
localtime, gmtime, asctime, timezone: convert	date and time to ASCII. ctime,	ctime(3)
touch: update	date last modified of a file.	touch(1)
idate, itime: return	date or time in numerical form.	idate(3F)
	date: print and set the date	date(1)
	date: print and set the date	date(1)
dbm delete, dbm firstkey, dbm nextkey, dbm error,	dbm clearerr: data base subroutines. /dbm store,	ndbm(3)
dbm firstkey, dbm nextkey, dbm error,/ dbm open,	dbm close, dbm fetch, dbm store, dbm_delete,	ndbm(3)
dbm open, dbm close, dbm fetch, dbm store,	dbm delete, dbm firstkey, dbm nextkey, dbm error,/	ndbm(3)
/dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey,	dbm_error, dbm_clearerr: data base subroutines	ndbm(3)
	doni_crror, doni_ckarerr. data base subrodines	3.5
	dhm faich dhm etora dhm dalata dhm firetkay	ndhm(3)
dbm_nextkey, dbm_error,/ dbm_open, dbm_close,	dbm_fetch, dbm_store, dbm_delete, dbm_firstkey,	ndbm(3)
data/ /dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr:	ndbm(3)
data/ /dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines.	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey:	ndbm(3) dbm(3X)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey,	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/	ndbm(3) dbm(3X) ndbm(3)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error,/	dbm firstkey, dbm nextkey, dbm error, dbm clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm nextkey, dbm error, dbm clearerr: data base/ dbm open, dbm close, dbm fetch, dbm store,	ndbm(3) dbm(3X) ndbm(3) ndbm(3)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey,	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, .	ndbm(3) dbm(3X) ndbm(3) ndbm(3)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error,/	dbm firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error,/	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, .	ndbm(3) dbm(3X) ndbm(3) ndbm(3)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error,/	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error,/	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error,/	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx:	dbm firstkey, dbm nextkey, dbm error, dbm clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm nextkey, dbm error, dbm clearerr: data base/ dbm open, dbm close, dbm fetch, dbm store, dbm store, dbm delete, dbm firstkey, dbm nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx symbol table information. dbx symbol table information.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx:	dbm firstkey, dbm nextkey, dbm error, dbm clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm nextkey, dbm error, dbm clearerr: data base/ dbm open, dbm close, dbm fetch, dbm store, dbm store, dbm delete, dbm firstkey, dbm nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx symbol table information. dbx symbol table information.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx:	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dbx(5)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx:	dbm firstkey, dbm nextkey, dbm error, dbm clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm nextkey, dbm error, dbm clearerr: data base/ dbm open, dbm close, dbm fetch, dbm store, dbm store, dbm delete, dbm firstkey, dbm nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx debugger. dbx symbol table information. dbx symbol table information. dc: desk calculator.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dbx(5) dbx(5)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx:	dbm firstkey, dbm nextkey, dbm error, dbm clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm nextkey, dbm error, dbm clearerr: data base/ dbm open, dbm close, dbm fetch, dbm store, dbm store, dbm delete, dbm firstkey, dbm nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx symbol table information. dbx symbol table information. dbx symbol table information. dc: desk calculator. dc: desk calculator.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dbx(5) dc(1) dc(1)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx: dbx:	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dbx symbol table information. dc: desk calculator. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx: dbx:	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dbx symbol table information. dc: desk calculator. dc: desk calculator. dc: desk calculator. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx: dbx:	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debug: debugger for the IBM RT PC.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx symbol table information. dc: desugger. dbx symbol table information. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debug: debugger for the IBM RT PC.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx symbol table information. dc: debugger. dbx symbol table information. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger. debugger. debugger.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm firstkey, dbm nextkey, dbm error, dbm clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm nextkey, dbm error, dbm clearerr: data base/ dbm open, dbm close, dbm fetch, dbm store, dbm store, dbm delete, dbm firstkey, dbm nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx: debugger. dbx symbol table information. dc: desk calculator. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debug: debugger for the IBM RT PC. debugger. debugger.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1) adb(1) dbx(1)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dc: desk calculator. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger. debugger. debugger. debugger. debugger. debugger.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) dbx(1) dbx(1)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx: dbx: dbx: dbx: dbx: pdx: pascal	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dbx symbol table information. dc: desk calculator. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger. debugger. debugger. debugger. debugger. debugger. debugger.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) dbx(1) dbx(1)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx: dbx: dbx: dbx: dbx: dbx: pdx: pascal debug:	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dc: desk calculator. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) ndbm(5) dbx(5) dbx(5) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) dbx(1) dbx(1) dbx(1) dbx(1) dbx(1) dbx(1)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx symbol table information. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger. debugger for the IBM RT PC. debugger symbol table.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) debug(8) makesym(8)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx: dbx: dbx: dbx: pdx: pascal debug: makesym: make pcl:	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx symbol table information. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger. debugger for the IBM RT PC. debugger symbol table. DEC CSS PCL-11 B Network Interface.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(5) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) dbx(1) dbx(1) dbx(1) dbx(1) pdx(1) debug(8) makesym(8) pcl(4)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm firstkey, dbm nextkey, dbm error, dbm clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm nextkey, dbm error, dbm clearerr: data base/ dbm open, dbm close, dbm fetch, dbm store, dbm store, dbm delete, dbm firstkey, dbm nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx: debugger. dbx: debugger. dbx symbol table information. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger. debugger. debugger. debugger. debugger. debugger. debugger. debugger for the IBM RT PC. debugger. debugger for the IBM RT PC. debugger symbol table. DEC CSS PCL-11 B Network Interface. DEC DEQNA Q-bus 10 Mb/s Ethernet interface.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(5) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) dbx(1) dbx(1) debug(8) makesym(8) pcl(4) qe(4)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm firstkey, dbm nextkey, dbm error, dbm clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm nextkey, dbm error, dbm clearerr: data base/ dbm open, dbm close, dbm fetch, dbm store, dbm store, dbm delete, dbm firstkey, dbm nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dbx symbol table information. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) dbx(1) dbx(1) dbx(1) pdx(1) debug(8) makesym(8) pcl(4) qe(4) de(4)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dc: desk calculator. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger. debugger symbol table. DEC CSS PCL-11 B Network Interface. DEC DEUNA 10 Mb/s Ethernet interface. DEC DEUNA 10 Mb/s Ethernet interface.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) dbx(1) dbx(1) debug(8) makesym(8) pcl(4) qe(4)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm firstkey, dbm nextkey, dbm error, dbm clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm nextkey, dbm error, dbm clearerr: data base/ dbm open, dbm close, dbm fetch, dbm store, dbm store, dbm delete, dbm firstkey, dbm nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dbx symbol table information. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) dbx(1) dbx(1) dbx(1) pdx(1) debug(8) makesym(8) pcl(4) qe(4) de(4)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete,	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dc: desk calculator. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger. debugger symbol table. DEC CSS PCL-11 B Network Interface. DEC DEUNA 10 Mb/s Ethernet interface. DEC DEUNA 10 Mb/s Ethernet interface.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(1) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) dbx(1) dbx(1) dbx(1) debug(8) makesym(8) pcl(4) qe(4) de(4) de(4)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx: dbx: dbx: dbx: dbx: pdx: pascal debug: makesym: make pcl: qe: de: device. dmc: css: rx:	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dc: desk calculator. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) dbx(1) dbx(1) pdx(1) debug(8) makesym(8) pcl(4) qe(4) de(4) de(4) de(4) druc(4) css(4) rx(4)
data//dbm_close, dbm_fetch, dbm_store, dbm_delete, data base subroutines. /dbm_fetch, dbm_store, dbm_delete, dbm_firstkey, dbm_delete, dbm_firstkey, dbm_nextkey, dbm_error./ dbm_error./ dbm_open, dbm_close, dbm_fetch, dbx: dbx: dbx: dbx: dbx: pdx: pascal debug: makesym: make pcl: qe: de: device. dmc: css: rx: bad144: read/write	dbm_firstkey, dbm_nextkey, dbm_error, dbm_clearerr: dbminit, fetch, store, delete, firstkey, nextkey: dbm_nextkey, dbm_error, dbm_clearerr: data base/ dbm_open, dbm_close, dbm_fetch, dbm_store, dbm_store, dbm_delete, dbm_firstkey, dbm_nextkey, dbx: dbx symbol table information. dbx: dbx symbol table information. dbx: debugger. dbx: debugger. dbx symbol table information. dc: desk calculator. dc: desk calculator. dc: desk calculator. dcheck: file system directory consistency check. dd: convert and copy a file. ddn: DDN Standard Mode X.25 IMP interface. DDN Standard Mode X.25 IMP interface. de: DEC DEUNA 10 Mb/s Ethernet interface. debugger. debugger. debugger. debugger. debugger. debugger. debugger. debugger. debugger. debugger symbol table. DEC CSS PCL-11 B Network Interface. DEC DEUNA 10 Mb/s Ethernet interface. DEC DEUNA 10 Mb/s Ethernet interface. DEC DEUNA 7-bus 10 Mb/s Ethernet interface. DEC DEUNA 10 Mb/s Ethernet interface.	ndbm(3) dbm(3X) ndbm(3) ndbm(3) ndbm(3) ndbm(3) dbx(5) dbx(5) dbx(1) dbx(5) dbx(5) dc(1) dcheck(8) dd(1) ddn(4) de(4) debug(8) adb(1) adb(1) dbx(1) dbx(1) dbx(1) dbx(1) dbx(1) dcheck(8) dd(1) dcheck(8) dcheck(8) dcheck(8) dcheck(8) dcheck(8) dcheck(9) dcheck(8) dcheck(1) dcheck

od: octal,	decimal, hex, ascii dump.	· • • · · ·
XMenu - X	Deck of cards Menu System	
tp:	DEC/mag tape formats	tp(5)
cc:	default C compiler	cc(1)
	default: catchall clause in switch	csh(1)
chdir: change	default directory.	chdir(3F)
diskpart: calculate	default disk partition sizes	diskpart(8)
diskpart: calculate	default disk partition sizes	diskpart(8)
kbdemul:	default keyboard emulator	kbdemul(4)
vgrindefs: vgrind's language	definition data base	vgrindefs(5)
stty, gtty: set and get terminal state	(defunct)	stty(3C)
close:	delete a descriptor	
dbminit, fetch, store,	delete, firstkey, nextkey: data base subroutines	
dosread: read, write, dir,	delete on PC-DOS diskette	
tail:	deliver the last part of a file	
mesg: permit or	· .	* * .
.		
tset: terminal	dependent initialization.	
qe: DEC	DEQNA Q-bus 10 Mb/s Ethernet interface	
constructs.	deroff: remove nroff, troff, tbl and eqn	
crypt, setkey, encrypt:	DES encryption	
whatis:	describe what a command is	
getdiskbyname: get disk	description by its name.	. ,
disktab: disk	description file.	
L-devices: UUCP device	description file.	L-devices(5)
L.sys: UUCP remote host	description file.	* * *
remote: remote host	description file.	remote(5)
close: delete a	descriptor	close(2)
dup, dup2: duplicate a	descriptor	dup(2)
getfstype, setfsent, endfsent: get file system	descriptor file entry. /getfsspec, getfsfile,	
getdtablesize: get	descriptor table size	
de:	desk calculator.	•
access:	determine accessibility of a file	` '
access:	determine accessibility of file	, , ,
file:	determine file type	
de: DEC	DEUNA 10 Mb/s Ethernet interface	
DEC DMC-11/DMR-11 point-to-point communications	device. dmc:	dmc(4)
drum: paging	device	
fold: fold long lines for finite width output	device.	
ioctl: control		fold(1)
L-devices: UUCP		
swapon: add a swap	device description file	
•		
swapon: specify additional	device for paging and swapping	
ik: Ikonas frame buffer, graphics		3.5
ps: Evans and Sutherland Picture System 2 graphics tb: line discipline for digitizing	device interface	*. *. /
		` '
tb: line discipline for digitizing	devices.	
	df: disk free.	, ,
flmin, flmax, ffrac, dflmin, dflmax,	dffrac, inmax: return extreme values	
flmin, flmax, ffrac, dflmin,	dsimax, dsfrac, inmax: return extreme values	
values. fimin, fimax, ffrac,	dflmin, dflmax, dffrac, inmax: return extreme	, ,
	dh: DII-11/DM-11 communications multiplexer	dh(4)
dh:	DH-11/DM-11 communications multiplexer	dh(4)
:	dhu: DIIU-11 communications multiplexer	dhu(4)
dhu:	DHU-11 communications multiplexer	dhu(4)
dmesg: collect system	diagnostic messages to form error log	dmesg(8)
autoconf:	diagnostics from the autoconfiguration code	autoconf(4)
autoconf:	diagnostics from the autoconfiguration code	autoconf(4)
explain: print wordy sentences; thesaurus for	diction. diction,	diction(1)
for diction.	diction, explain: print wordy sentences; thesaurus	diction(1)
	diff: differential file and directory comparator	diff(1) `
	diff3: 3-way differential file comparison.	
diff:	differential file and directory comparator	
diff3: 3-way	differential file comparison.	diff3(1)
xclock - X Window System, analog /	digital clock.	. '. '
tb: line discipline for	digitizing devices.	tb(4)
tb: line discipline for	digitizing devices.	. 1.1
dosread: read, write,	dir, delete on PC-DOS diskette.	
doploud, 1984, Wille,	dir: format of directories.	
fpa:	direct interface to floating point accelerator.	
dir: format of	directories	
rm, rmdir: remove (unlink) files or	directories	rm(1)
sticky: persistent text and append-only	directories	sticky(8)
rmdir, rm: remove (unlink)	directories or files.	rmdir(1)
cd: change working	directory.	d(1)
chdir: change current working	directory.	chdir(2)
chroot: change current working chroot:	directory.	
Cili Oot. Change 100t	· uncounty.	- Citi 604(2)

cd: change	directory	csh(1)
chdir: change	directory	csh(1)
ls: list contents of	directory	ls(1)
mkdir: make a	directory	mkdir(1)
chdir: change default getcwd: get pathname of current working	directory	chdir(3F) getcwd(3F)
scandir, alphasort: scan a	directory.	scandir(3)
uuclean: uucp spool	directory clean-up.	uuclean(8C)
diff: differential file and	directory comparator	diff(1)
dcheck: file system	directory consistency check	dcheck(8)
unlink: remove	directory entry.	unlink(2)
unlink: remove a	directory entry.	unlink(3F)
mkdir: make a	directory file.	mkdir(2)
rmdir: remove a mklost + found: make a lost + found	directory file	rmdir(2)
pwd: working	directory for fsck	mklost + found(8) pwd(1)
readdir, telldir, seekdir, rewinddir, closedir:	directory operations. opendir,	directory(3)
getwd: get current working	directory pathname.	getwd(3)
whois: DARPA Internet user name	directory service.	whois(1)
popd: pop shell	directory stack	csh(1)
pushd: push shell	directory stack	csh(1)
quota: display	disc usage and limits	quota(1)
unhash:	discard command hash table	csh(1)
unset:	discard shell variables	csh(1)
ap: asynchronous data mode protocol line tb: line	discipline	ap(4)
tb: line	discipline for digitizing devices	tb(4) tb(4)
(obsolete). bk: line	discipline for machine-machine communication	bk(4)
disk: format of reserved areas of the hard	disk	disk(4)
synchronize a file's in-core state with that on	disk. fsync:	fsync(2)
hk: RK6-11/RK06 and RK07 moving head	disk	hk(4)
uda: UDA-50	disk controller interface	uda(4)
getdiskbyname: get	disk description by its name	getdisk(3)
disktab:	disk description file.	disktab(5)
df:	disk: format of reserved areas of the hard disk disk free	disk(4)
hd: hard	disk interface.	df(1) hd(4)
hp: MASSBUS	disk interface	hp(4)
rx: DEC RX02 floppy	disk interface.	rx(4)
rvdcopy: copy contents of one RVD	disk pack to another.	rvdcopy(8)
format: how to format	disk packs	format(8V)
diskpart: calculate default	disk partition sizes.	diskpart(8)
diskpart: calculate default	disk partition sizes.	diskpart(8)
rvd: Remote Virtual	Disk protocol	rvd(4p)
quota: manipulate newvd: create a new filesystem on a Remote Virtual	disk quotas	quota(2)
vdspind: spin up or spin down a Remote Virtual	Disk (RVD)	newvd(8) vdspin(2)
vddb: Remote Virtual	Disk (RVD) data base manager	vddb(8)
rvddown: force spindown of a Remote Virtual	Disk (RVD) pack	rvddown(8)
spinup, spindown: spin up/down Remote Virtual	Disk (RVD) pack	spinup(8)
rvdexch: exchange names of two Remote Virtual	Disk (RVD) packs	rvdexch(8)
rvdflush: spindown client's Remote Virtual	Disk (RVD) packs	rvdflush(8)
savephys: back up and restore Remote Virtual	Disk (RVD) packs to and from tape. /zaprvd,	savervd(8)
rvdchlog: change logging level of Remote Virtual rvdgetm: get operations message from Remote Virtual	Disk (RVD) server.	rvdchlog(8)
rvdsend - send control stream to Remote Virtual	Disk (RVD) server	rvdgetm(8) rvdsend(8)
rvdsetm: set operations message on Remote Virtual	Disk (RVD) server.	rvdsetm(8)
rvdshow: show connections to Remote Virtual	Disk (RVD) server	rvdshow(8)
rvdshut: force shutdown of Remote Virtual	Disk (RVD) server	rvdshut(8)
rvddb: Remote Virtual	Disk (RVD) server configuration table	rvddb(5)
rvdsrv: Remote Virtual	Disk (RVD) server daemon	rvdsrv(8)
rvdlog: cause Remote Virtual	Disk (RVD) server to log statistics	rvdlog(8)
vdstats: acquire client Remote Virtual vdstats: list client Remote Virtual	Disk (RVD) statistics	vdstats(2)
up, down: client Remote Virtual	Disk (RVD) statistics	vdstats(8) up(1)
drtest: standalone	disk test program.	drtest(8)
scsiformat: format the IBM 9332	disk unit.	scsiformat(8c)
du: summarize	disk usage.	du(1)
dosread: read, write, dir, delete on PC-DOS	diskette	dosréad(1)
fd:	diskette interface	fd(4)
fdformat: format	diskettes	fdformat(8r)
flcopy: copier for	diskettes	flcopy(8r)
	diskpart: calculate default disk partition sizes	diskpart(8)
reboot/halt the system without checking the	diskpart: calculate default disk partition sizes disks. fastboot, fasthalt:	diskpart(8) fastboot(8)
format: format hard	disks	format(8r)
rottilat iottilat iialu		. Or mac(or)

Comments Comments Comments	dt.d	C(01/)
rxformat: format floppy information about client Remote Virtual	disks. Disks (RVDs). /etc/rvd/rvdtab:	rxformat(8V) rvdtab(5)
(SCSI) Adapter. sc: IBM 9332	disks using the IBM Small Computer System Interface	sc(4)
` · ·	disktab: disk description file.	disktab(5)
mount, umount: mount and	dismount file system.	mount(8)
error: analyze and error: analyze and	disperse compiler error messages	error(1) error(1)
the IBM Academic Information Systems experimental	display. aedemul: graphics interfaces for	aedemul(4)
xload - X window system load average	display	xload(1)
ibm8604. ibm8514: IBM 8514/A	Display adapter for the ibm8503, ibm8513, ibm8514,	ibm8514(4)
arp: address resolution	display and control.	arp(8C)
bitprt: capture the image on a bitmap gprof:	display and print it on an IBM printer	bitprt(1) gprof(1)
aedjournal:	display commands in a log file	aedjournal(1)
VI_EnCur, VI_DisCur, VI_PosnCur: control the	display cursor. VI_MDefnCur, VI_FDefnCur,	cursor(3G)
VI_MRead, VI_FRead: read	display data.	read(3G)
quota: vi: screen oriented (visual)	display disc usage and limits	quota(1) vi(1)
umask: change or	display file creation mask	csh(1)
intro: introduction to	display graphics subroutines	intro(3G)
cons: keyboard and console	display interface	cons(4)
ibm5081, mpel – IBM 5081 Mega Pel	Display interface	ibm5081(4)
ibm5151, mono: IBM 5151 Monochrome ibm5154, ega: IBM 5154 Enhanced Graphics	Display interface	ibm5151(4) ibm5154(4)
apa8: IBM 6153 Advanced Monochrome Graphics	Display interface. ibm6153,	ibm6153(4)
ibm6154, apa8c: IBM 6154 Advanced Color Graphics	Display interface	ibm6154(4)
apa16: IBM 6155 Extended Monochrome Graphics	Display interface. ibm6155,	ibm6155(4)
aed: IBM Academic Information Systems experimental	display interface. ibmaed,	ibmaed(4) dumpaed(1)
dumpaed: dump aed dumpapal6: dump apal6	display memory as a binary file	dumpapa16(1)
dumpapa8: dump apa8	display memory as a binary file	dumpapa8(1)
dumpapa8c: dump apa8c	display memory as a binary file	dumpapa8c(1)
prof:	display profile data	prof(1)
setscreen: control consoles: utility database of	display screen access	setscreen(8) consoles(5)
IBM Academic Information Systems experimental	display self-tests. aedtest:	aedtest(8)
systat:	display system statistics on a crt	systat(1)
sysline:	display system status on status line of a terminal	sysline(1)
xfd - X window system font xlsfonts - X window system font list	displayer	xfd(1) xlsfonts(1)
xprop - X Window System property	displayer	xprop(1)
hypot, cabs: Euclidean	distance, complex absolute value	hypot(3M)
rdist: remote file	distribution program.	rdist(1)
communications device.	dmc: DEC DMC-11/DMR-11 point-to-point	dmc(4)
dmc: DEC error log.	DMC-11/DMR-11 point-to-point communications device. dmesg: collect system diagnostic messages to form	dmc(4) dmesg(8)
citor log.	dmf: DMF-32, terminal multiplexor	dmf(4)
dmf:	DMF-32, terminal multiplexor	dmf(4)
	dmz: DMZ-32 terminal multiplexor	dmz(4)
dmz:	DMZ-32 terminal multiplexor	dmz(4) dn(4)
dn:	DN-11 autocall unit interface.	dn(4)
res_mkquery, res_send, res_init,	dn_comp, dn_expand: resolver routines	resolver(3)
res_mkquery, res_send, res_init, dn_comp,	dn_expand: resolver routines	resolver(3)
style: analyze surface characteristics of a	document.	style(1)
refer: find and insert literature references in w: who is on and what they are	documents	refer(1) w(1)
named: Internet	domain name server.	named(8)
unlog: tell Venus you are	done with vice.	unlog(1)
diskette.	dosread: read, write, dir, delete on PC-DOS	dosread(1)
vdabort: abort and spin vdspin, vdspind: spin up or spin	down a drive	vdabort(8) vdspin(2)
up,	down: client Remote Virtual Disk (RVD) utilities	up(1)
shutdown: shut	down part of a full-duplex connection	shutdown(2)
shutdown: close	down the system at a given time	shutdown(8)
rand, random,	drand, irand: return random values drandm, irandm: better random number generator	rand(3F) random(3F)
VI Circle:	draw a circle.	circle(3G)
graph:	draw a graph	graph(1G)
VI_ALine, VI_RLine:	draw a line.	line(3G)
VI_String: VI_MImage, VI_FImage:	draw a string	string(3G) image(3G)
pic: troff preprocessor for	drawing simple pictures.	pic(1)
exponent manipulations. copysign,	drem, finite, logb, scalb: copysign, remainder,	ieee(3M)
isnan, unordered, finite, infinity,/ copysign,	drem, logb, scalb, rint, classdouble, classfloat,	ieee(3)
vdabort: abort and spin down a	drive	vdabort(8)

ut: UNIBUS TU45 tri-density tape	drive interface.	ut(4)
pty: pseudo terminal	driver	pty(4)
•	drtest: standalone disk test program.	
atima	drum: paging device.	` '
etime,	dtime: return elapsed execution time	etime(3F) du(1)
dump: incremental file system	dump.	dump(8)
od: octal, decimal, hex, ascii	dump	od(1)
xpr: print X window	dump	xpr(1)
rdump: file system	dump across the network.	rdump(8C)
rrestore: restore a file system dumpaed:	dump across the network	rrestore(8C)
dumpapa16:	dump apa16 display memory as a binary file	dumpaed(1) dumpapa16(1)
dumpapa8:	dump apa8 display memory as a binary file	dumpapa8(1)
dumpapa8c:	dump apa8c display memory as a binary file	dumpapa8c(1)
nfabort:	dump core and log it in a notesfile	nfabort(3)
4	dump, dumpdates: incremental dump format	dump(5)
dumpfs: dump, dumpdates: incremental	dump file system information	dumpfs(8)
landump:	dump IBM Token-Ring Personal Computer Adapter.	dump(5) landump(8r)
missinp.	dump: incremental file system dump	dump(8)
savecore: save a core	dump of the operating system	savecore(8)
kgmon: generate a	dump of the operating system's profile buffers	kgmon(8)
	dumpaed: dump aed display memory as a binary file	dumpaed(1)
file.	dumpapa16: dump apa16 display memory as a binary .	dumpapa16(1)
file. file,	dumpapa8: dump apa8 display memory as a binary dumpapa8c: dump apa8c display memory as a binary	dumpapa8(1) dumpapa8c(1)
dump,	dumpdates: incremental dump format	dump(5)
xwd - X Window System, window image	dumper	xwd(1)
,	dumpfs: dump file system information	dumpfs(8)
	dup, dup2: duplicate a descriptor	dup(2)
dup,	dup2: duplicate a descriptor	dup(2)
dup, dup2:	duplicate a descriptor	dup(2)
dz:	• • • • • • • • • • • • • • • • • • •	dz(4) dz(4)
52.	ec: 3Com 10 Mb/s Ethernet interface	ec(4)
echo:	echo arguments	csh(1)
echo:	echo arguments.	echo(1)
	echo: echo arguments.	csh(1)
ping: send ICMP	echo: echo arguments	echo(1) ping(8)
ping, send icivit	ecvt, fcvt, gcvt: output conversion	ecvt(3)
	ed: text editor.	ed(1)
end, etext,	edata: last locations in program	end(3)
ex,	edit: text editor.	ex(1)
vipw:	edit the password file.	vipw(8)
edquota: ed: text	edit user quotas	edquota(8) ed(1)
ex, edit: text	editor.	ex(1)
ld: link	editor	ld(1)
ld: link	editor	ld(1)
sed: stream	editor.	sed(1)
vi: screen oriented (visual) display	editor based on ex	vi(1)
bitmap: bitmap a.out: assembler and link	editor for X window system	bitmap(1) a.out(5)
a.out: assembler and link	editor output.	a.out(5)
	edquota: edit user quotas.	edquota(8)
whoami: print	effective current user id.	whoami(1)
setregid: set real and	effective group ID	setregid(2)
setreuid: set real and	effective user ID's	setreuid(2)
vfork: spawn new process in a virtual memory	efficient way	vfork(2) efl(1)
ibm5154,	ega: IBM 5154 Enhanced Graphics Display interface.	ibm5154(4)
grep,	egrep, fgrep: search a file for a pattern	grep(1)
etime, dtime: return	elapsed execution time	etime(3F)
insque, remque: insert/remove	element from a queue.	insque(3)
soelim:	eliminate .so's from nroff input.	soelim(1)
bufemul: kernel buffering	else: alternative commands	csh(1)
getspemulator: return address of the floating-point	emulator	bufemul(4) getfpemulator(2)
ibmemul: IBM 3101	emulator.	ibmemul(4)
kbdemul: default keyboard	emulator.	kbdemul(4)
stdemul: standard output	emulator	stdemul(4)
xterm: X window system terminal	emulator	xterm(1)
xemul: X input Xtty: routines to provide terminal	emulator for queuing keyboard and mouse events emulator windows	xemul(4)
Aug. routiles to provide terminal	Circumtor Willidows	xtty(3X)

	en: Xerox 3 Mb/s Ethernet interface	en(4)
setquota:	enable/disable quotas on a file system.	setquota(2)
nsip: software network interface	encapsulating ns packets in ip packets	nsip(4)
uuencode: format of an	encoded uuencode file	uuencode(5)
crypt:	encode/decode	crypt(1)
crypt, setkey,	encrypt: DFS encryption	crypt(3)
crypt, setkey, encrypt: DES	encryption	crypt(3)
makekey: generate	encryption key.	makekey(8)
	end, etext, edata: last locations in program	end(3)
sccs: front	end for the SCCS subsystem	sccs(1)
logout:	end session	csh(1)
	end: terminate loop.	csh(1)
/getfsspec, getfsfile, getfstype, setfsent,	endfsent: get file system descriptor file entry	getfsent(3)
getgrent, getgrgid, getgrnam, setgrent,	endgrent: get group file entry	getgrent(3)
gethostbyaddr, gethostent, sethostent,	endhostent: get network host entry. gethostbyname,	gethostbyname(3N)
	endif: terminate conditional	csh(1)
getnetent, getnetbyaddr, getnetbyname, setnetent,	endnetent: get network entry.	getnetent(3N)
socket: create an	endpoint for communication.	socket(2)
getprotobynumber, getprotobyname, setprotoent,	endprotoent: get protocol entry. getprotoent,	getprotoent(3N)
getpwent, getpwuid, getpwnam, setpwent,	endpwent, setpwfile: get password file entry	getpwent(3)
getservbyport, getservbyname, setservent,	endservent: get service entry. getservent,	getservent(3N)
Berner coppored Berner community control county	endsw: terminate switch.	csh(1)
getttyent, getttynam, setttyent,	endttyent: get ttys file entry.	getttyent(3)
gettyent, gettynam, settyent, getusershell, setusershell,		
	endusershell: get legal user shells	getusershell(3)
ibm5154, ega: IBM 5154	Enhanced Graphics Display interface	ibm5154(4)
xsend, xget,	enroll: secret mail	xsend(1)
nlist: get	entries from name list.	nlist(3)
logger: make	entries in the system log.	logger(1)
unlink: remove directory	entry	unlink(2)
setssent, endssent: get file system descriptor file	entry. getssent, getssspec, getssfile, getsstype,	getfsent(3)
getgrnam, setgrent, endgrent: get group file	entry. getgrent, getgrgid,	getgrent(3)
sethostent, endhostent: get network host	entry. gethostbyname, gethostbyaddr, gethostent,	gethostbyname(3N)
getnetbyname, setnetent, endnetent: get network	entry. getnetent, getnetbyaddr,	getnetent(3N)
setprotoent, endprotoent: get protocol	entry. /getprotobynumber, getprotobyname,	getprotoent(3N)
setpwent, endpwent, setpwfile: get password file	entry. getpwent, getpwuid, getpwnam,	getpwent(3)
getservbyname, setservent, endservent: get service	entry. getservent, getservbyport,	getservent(3N)
getttynam, setttyent, endttyent: get ttys file	entry. getttyent,	getttyent(3)
unlink: remove a directory	entry.	unlink(3F)
execv, execle, execlp, execvp, exec, execve, exect,	environ: execute a file. execl,	execl(3)
setenv: set variable in	environment	csh(1)
printenv: print out the	environment	printenv(1)
window: window	environment	window(1)
getenv: value for	environment name	getenv(3)
unsetenv: remove	environment variables	csh(1)
getenv: get value of	environment variables	getenv(3F)
deroff: remove nroff, troff, tbl and	eqn constructs	deroff(1)
	eqn, neqn, checkeq: typeset mathematics	eqn(1)
linemod, space, closepl: graphics/ plot: openpl,	erase, label, line, circle, arc, move, cont, point,	plot(3X)
	erf, erfc: error functions.	erf(3M)
erf.	erfc: error functions.	erf(3M)
messages.	error: analyze and disperse compiler error	error(1)
messages.	error: analyze and disperse compiler error	error(1)
erf, erfc:	error functions.	erf(3M)
dmesg: collect system diagnostic messages to form	error log.	dmesg(8)
mkstr: create an	error message file by massaging C source.	mkstr(1)
	• • •	-1.1
error: analyze and disperse compiler	error messages	error(1)
error: analyze and disperse compiler	error messages.	error(1)
perror, sys_errlist, sys_nerr: system	error messages.	perror(3)
perror, gerror, ierrno: get system	error messages.	perror(3F)
intro: introduction to system calls and	error numbers.	intro(2)
intro: introduction to system calls and	error numbers.	intro(2)
spell, spellin, spellout: find spelling	errors	spell(1)
traper: trap arithmetic	errors.	traper(3F)
Virtual Disks (RVDs).	/etc/rvd/rvdtab: information about client Remote	rvdtab(5)
end,	etext, edata: last locations in program.	end(3)
un: IBM RT PC Baseband Adapter for use with	Ethernet.	un(4)
de: DEC DEUNA 10 Mb/s	Ethernet interface.	de(4)
ec: 3Com 10 Mb/s	Ethernet interface.	ec(4)
en: Xerox 3 Mb/s	Ethernet interface	1
•		en(4)
ex: Excelan 10 Mb/s	Ethernet interface.	ex(4)
il: Interlan NI1010 10 Mb/s	Ethernet interface	il(4)
ix: Interlan Np100 10 Mb/s	Ethernet interface.	ix(4)
np: Interlan Np100 10 Mb/s	Ethernet interface.	np(4)
qe: DEC DEQNA Q-bus 10 Mb/s	Ethernet interface	qe(4)
· · · · · · · · · · · · · · · · · · ·	etime, dtime: return elapsed execution time	etime(3F)
hypot, cabs:	Euclidean distance, complex absolute value	hypot(3M)

11 M 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		1.44
/if, while, :, ., break, continue, cd,	eval, exec, exit, export, login, read, readonly,/	sh(1)
expr:	eval: re-evaluate shell data	csh(1) expr(1)
device interface. ps:	Evans and Sutherland Picture System 2 graphics	ps(4)
history: print history	event list.	csh(1)
X input emulator for queuing keyboard and mouse	events. xemul:	xemul(4)
screen oriented (visual) display editor based on	ex. vi:	vi(1)
	ex, edit: text editor.	ex(1)
1, 1	ex: Excelan 10 Mb/s Ethernet interface	ex(4)
lpq: spool queue	examination program.	Ipq(1)
ex: packs. rvdexch:	Excelan 10 Mb/s Ethernet interface exchange names of two Remote Virtual Disk (RVD) .	ex(4) rvdexch(8)
exect, execv, execte, exectp, execvp,	exec, execve, exect, environ: execute a file	exect(3)
/while, :, ., break, continue, cd, eval,	exec, exit, export, login, read, readonly, set,/	sh(1)
	exec: overlay shell with specified command	csh(1)
exect, environ: execute a file.	execl, execv, execle, execlp, execvp, exec, execve,	execl(3)
environ: execute a file. execl, execv,	execle, execlp, execvp, exec, execve, exect,	execl(3)
execute a file. exect, exect, execte,	execlp, execvp, exec, execve, exect, environ:	exect(3)
execl, execv, execle, execlp, execvp, exec, execve, execve:	exect, environ: execute a file	execl(3) execve(2)
execlp, execvp, exec, execve, exect, environ:	execute a file. execl, execv, execle,	exect(3)
alarm:	execute a subroutine after a specified time	alarm(3F)
system:	execute a UNIX command	system(3F)
repeat:	execute command repeatedly	csh(1)
at:	execute commands at a later time	at(1)
aedrunner:	execute graphics commands in a log file	aedrunner(1)
lastcomm: show last commands acct;	executed in reverse order	lastcomm(1)
uuxqt: UUCP	execution file interpreter.	acct(5) uuxqt(8C)
sleep: suspend	execution for an interval.	sleep(1)
sleep: suspend	execution for an interval.	sleep(3F)
sleep: suspend	execution for interval	sleep(3)
usleep: suspend	execution for interval	usleep(3)
monitor, monstartup, moncontrol: prepare	execution profile.	monitor(3)
pxp: Pascal	execution profiler	pxp(1)
rexecd: remote etime, dtime: return elapsed	execution server	rexecd(8C) etime(3F)
profil:	execution time profile	profil(2)
pix: Pascal interpreter and	executor.	pix(1)
environ: execute a file. execl,	execv, execle, execlp, execvp, exec, execve, exect,	execl(3)
execl, execv, execle, execlp, execvp, exec,	execve, exect, environ: execute a file	execl(3)
Ct	execve: execute a file	execve(2)
file. execl, execv, execle, execlp, link: make a link to an	execvp, exec, execve, exect, environ: execute a existing file	execl(3) link(3F)
tunefs: tune up an	existing file system.	tunefs(8)
/:, ., break, continue, cd, eval, exec,	exit, export, login, read, readonly, set, shift,/	sh(1)
breaksw:	exit from switch.	csh(1)
	exit: leave shell.	csh(1)
12	_exit: terminate a process.	exit(2)
pending output.	exit: terminate a process after flushing any	exit(3) exit(3F)
break:	exit: terminate process with status	csh(1)
logarithm, power.	exp, expm1, log, log10, log1p, pow: exponential,	exp(3M)
glob: filename	expand argument list	csh(1)
compress, uncompress, zcat: compress and	expand data	compress(1)
expand, unexpand:	expand tabs to spaces, and vice versa	expand(1)
versa.	expand, unexpand: expand tabs to spaces, and vice	expand(1)
interfaces for the IBM Academic Information Systems ibmaed, aed: IBM Academic Information Systems	experimental display. aedemul: graphics experimental display interface	aedemul(4) ibmaed(4)
aedtest: IBM Academic Information Systems	experimental display self-tests.	aedtest(8)
diction. diction.	explain: print wordy sentences; thesaurus for	diction(1)
logarithm, power. exp,	expm1, log, log10, log1p, pow: exponential,	exp(3M)
frexp, ldexp, modf: split into mantissa and	exponent	frexp(3)
drem, finite, logb, scalb: copysign, remainder,	exponent manipulations. copysign,	ieee(3M)
exp, expm1, log, log10, log1p, pow:	exponential, logarithm, power	exp(3M)
/., break, continue, cd, eval, exec, exit,	export, login, read, readonly, set, shift, times,/ expr: evaluate arguments as an expression	sh(1)
expr: evaluate arguments as an	• •	expr(1) expr(1)
re_comp, re_exec: regular	expression handler.	regex(3)
addbib: create or	extend bibliographic database.	addbib(1)
efl:	Extended Fortran Language	efl(1)
ibm6155, apa16: IBM 6155	Extended Monochrome Graphics Display interface	ibm6155(4)
strings. xstr:	extract strings from C programs to implement shared .	xstr(1)
	f77: Fortran 77 compiler	f77(1)
ininit change	f77: FORTRAN 77 compiler	f77(1) ioinit(3F)
ionit. change	The state of the s	.onm(or)

libraries plot: openpl et al.:	f77 library interface to plot (3X)	plot(3F)
tclose, tread, twrite, trewin, tskipf, tstate: functions.	f77 tape I/O. topen,	topen(3F) floor(3M)
networking: introduction to networking	facilities.	intro(4N)
sigvec: software signal	facilities.	sig vec(2)
sigvec: software signal	facilities	sigvec(2)
signal: simplified software signal	facilities.	signal(3C)
malloc, free,	falloc: memory allocator	malloc(3F)
true,	false: provide truth values	true(1) false(1)
inet: Internet protocol	family.	inet(4F)
ns: Xerox Network Systems(tm) protocol	family.	ns(4F)
checking the disks.	fastboot, fasthalt: reboot/halt the system without	fastboot(8)
the disks. fastboot,	fasthalt: reboot/halt the system without checking	fastboot(8)
abort: generate a	fault.	abort(3)
trpfpe, fpecnt: trap and repair floating point	faults.	trpfpe(3F)
export, login,/sh, for, case, if, while, :, exit, export, login,/sh, for, case, if, while,	., break, continue, cd, eval, exec, exit,	sh(1) sh(1)
chmod.	fchmod: change mode of file.	chmod(2)
chown,	fchown: change owner and group of a file	chown(2)
	fclose, fflush: close or flush a stream	fclose(3S)
	fcntl: file control	fcntl(2)
ecvt,	fevt, gevt: output conversion	ecvt(3)
	fd: diskette interface.	fd(4)
	fdate: return date and time in an ASCII string	fdate(3F)
utility.	fdformat: format diskettes	fdformat(8r)
fopen, freopen,	fdopen: open a stream.	fdisk(8) fopen(3S)
ferror,	feof, clearerr, fileno: stream status inquiries.	ferror(3S)
inquiries.	ferror, feof, clearerr, fileno: stream status	ferror(3S)
subroutines. dbminit,	fetch, store, delete, firstkey, nextkey: data base	dbm(3X)
head: give first	few lines.	head(1)
fclose,	Mush: close or flush a stream.	fclose(3S)
extreme values. fimin, fimax,	ffrac, dflmin, dflmax, dffrac, inmax: return	fimin(3F)
bcopy, bcmp, bzero,	ffs: bit and byte string operations	bstring(3) csh(1)
getc,	fg: bring job into foreground	getc(3F)
getc, getchar,	fgetc, getw: get character or word from stream	getc(3S)
	fgets: get a string from a stream	gets(3S)
gets,	-B B	
grep, egrep,	fgrep: search a file for a pattern	grep(1)
grep, egrep, locate a program file including aliases and paths	fgrep: search a file for a pattern	grep(1) which(1)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of	fgrep: search a file for a pattern	grep(1) which(1) access(2)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting	fgrep: search a file for a pattern	grep(1) which(1) access(2) acct(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log	fgrep: search a file for a pattern	grep(1) which(1) access(2) acct(5) aedjournal(1)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting	fgrep: search a file for a pattern	grep(1) which(1) access(2) acct(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, schmod: change mode of chown, schown: change owner and group of a	fgrep: search a file for a pattern. (csh only). which: file. file. file. file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpaed(1)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa16 display memory as a binary	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpaed(1) dumpapa16(1)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa16 display memory as a binary dumpapa8: dump apa8 display memory as a binary	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpaed(1) dumpapa16(1) dumpapa8(1)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa16 display memory as a binary	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpaed(1) dumpapa16(1) dumpapa8(1) dumpapa8(1) dumpapa8c(1)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa8 display memory as a binary dumpapa8: dump apa8 display memory as a binary dumpapa8c: dump apa8 display memory as a binary	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpaed(1) dumpapa16(1) dumpapa8(1)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa8 display memory as a binary dumpapa8c: dump apa8e display memory as a binary dumpapa8c: dump apa8e display memory as a binary execve: execute a	fgrep: search a file for a pattern. (csh only). which: file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpapa16(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) execve(2) flock(2) fpr(1)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa8 display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a binary flock: apply or remove an advisory lock on an open fpr: print Fortran group: group	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) dctags(1) dd(1) disktab(5) dumpapa16(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) execve(2) flock(2) fpr(1) group(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa8: dump apa8 display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a binary flock: apply or remove an advisory lock on an open fpr: print Fortran group: group L.aliases: UUCP hostname alias	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpapa16(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) execve(2) flock(2) fpr(1) group(5) L.aliases(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fohmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa8 display memory as a binary dumpapa8e: dump apa8 display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a contact and copy a flock: apply or remove an advisory lock on an open fpr: print Fortran group: group Laliases: UUCP hostname alias L.cmds: UUCP remote command permissions	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpapa16(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) cxecve(2) flock(2) flock(2) fpr(1) group(5) L.aliases(5) L.cmds(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa16 display memory as a binary dumpapa8: dump apa8 display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa9c display memory as a binary cecve: execute a flock: apply or remove an advisory lock on an open fpr: print Fortran group: group Laliases: UUCP hostname alias L.cmds: UUCP remote command permissions L-devices: UUCP device description	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpapa16(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) cxecve(2) flock(2) fpr(1) group(5) L.aliases(5) L.cmds(5) L-devices(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa16 display memory as a binary dumpapa8: dump apa8 display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a binary group execve: execute a flock: apply or remove an advisory lock on an open fpr: print Fortran group: group Laliases: UUCP hostname alias L.cmds: UUCP remote command permissions L-devices: UUCP device description L-dialcodes: UUCP phone number index	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpapa16(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) cxecve(2) flock(2) fpr(1) group(5) L.aliases(5) L.cmds(5) L-devices(5) L-dialcodes(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa16 display memory as a binary dumpapa8: dump apa8 display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa9c display memory as a binary cecve: execute a flock: apply or remove an advisory lock on an open fpr: print Fortran group: group Laliases: UUCP hostname alias L.cmds: UUCP remote command permissions L-devices: UUCP device description	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) execve(2) flock(2) fpr(1) group(5) L.aliases(5) L.cmds(5) L-devices(5) L-dialcodes(5) link(2)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image core: read commands from ctags: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa16 display memory as a binary dumpapa8e: dump apa8e display memory as a binary dumpapa8e: dump apa8e display memory as a binary dumpapa8c: dump apa8e display memory as a binary execve: execute a flock: apply or remove an advisory lock on an open fpr: print Fortran group: group Laliases: UUCP hostname alias L.cmds: UUCP remote command permissions L-devices: UUCP device description L-dialcodes: UUCP phone number index link: make a hard link to a	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpapa16(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) cxecve(2) flock(2) fpr(1) group(5) L.aliases(5) L.cmds(5) L-devices(5) L-dialcodes(5)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log chmod, fehmod: change mode of chown, fehown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image core: format of memory image core: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa8 display memory as a binary dumpapa8: dump apa8 display memory as a binary dumpapa8c: dump apa8c display memory as a binary flock: apply or remove an advisory lock on an open fpr: print Fortran group: group L.aliases: UUCP hostname alias L.cmds: UUCP remote command permissions L-devices: UUCP device description L-dialcodes: UUCP phone number index link: make a hard link to a L.sys: UUCP remote host description mkdir: make a directory mknod: make a special	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) dctags(1) dd(1) disktab(5) dumpapa16(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) cxecve(2) ffock(2) fpr(1) group(5) L.aliases(5) L.cmds(5) L-dialcodes(5) link(2) L.sys(5) mkdir(2) mknod(2)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log chmod, fehmod: change mode of chown, fehown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa86 display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8ci dumpapa8c dump apa8c display me	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpapa16(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) execve(2) flock(2) fpr(1) group(5) L.aliases(5) L.cmds(5) L.devices(5) L.devices(5) L.dialcodes(5) link(2) L.sys(5) mkdir(2) mknod(2) mknod(8)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa86 display memory as a binary dumpapa8c: dump apa8 display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a binary cexeve: execute a flock: apply or remove an advisory lock on an open fpr: print Fortran group: group Laliases: UUCP hostname alias L.cmds: UUCP remote command permissions L-devices: UUCP device description L-dialcodes: UUCP phone number index link: make a hard link to a L.sys: UUCP remote host description mkdir: make a directory mknod: make a special mknod: build special rebuild the data base for the mail aliases	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) dtags(1) dd(1) disktab(5) dumpapa16(1) dumpapa8(1)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa86 display memory as a binary dumpapa8e: dump apa8e display memory as a binary dumpapa8e: dump apa8e display memory as a binary dumpapa8e: dump apa8e: display memory as a binary dumpapa display m	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) ctags(1) dd(1) disktab(5) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) dumpapa8(1) Execve(2) flock(2) fpr(1) group(5) L.aliases(5) L.cmds(5) L.devices(5) L.devices(5) L-dialcodes(5) link(2) L.sys(5) mkdir(2) mknod(8) newaliases(1) open(2)
grep, egrep, locate a program file including aliases and paths access: determine accessibility of acct: execution accounting aedjournal: display commands in a log aedrunner: execute graphics commands in a log chmod, fchmod: change mode of chown, fchown: change owner and group of a colrm: remove columns from a core: format of memory image core: format of memory image creat: create a new source: read commands from ctags: create a tags dd: convert and copy a disktab: disk description dumpaed: dump aed display memory as a binary dumpapa16: dump apa86 display memory as a binary dumpapa8c: dump apa8 display memory as a binary dumpapa8c: dump apa8c display memory as a binary dumpapa8c: dump apa8c display memory as a binary cexeve: execute a flock: apply or remove an advisory lock on an open fpr: print Fortran group: group Laliases: UUCP hostname alias L.cmds: UUCP remote command permissions L-devices: UUCP device description L-dialcodes: UUCP phone number index link: make a hard link to a L.sys: UUCP remote host description mkdir: make a directory mknod: make a special mknod: build special rebuild the data base for the mail aliases	fgrep: search a file for a pattern. (csh only). which: file. file.	grep(1) which(1) access(2) acct(5) aedjournal(1) aedrunner(1) chmod(2) chown(2) colrm(1) core(5) core(5) creat(2) csh(1) dtags(1) dd(1) disktab(5) dumpapa16(1) dumpapa8(1)

	C1-	
remote: remote host description	file.	remote(5)
rename: change the name of a	file.	rename(2)
resolver configuration	file	resolver(5)
rev: reverse lines of a	file.	rev(1)
rmdir: remove a directory	file.	1. 1
		rmdir(2)
size: size of an object	file	size(1)
the printable strings in a object, or other binary,	file. strings: find	strings(1)
sum: sum and count blocks in a	file.	sum(1)
symlink: make symbolic link to a	file.	2. 1
		symlink(2)
tail: deliver the last part of a	file	tail(1)
touch: update date last modified of a	file	touch(1)
uniq: report repeated lines in a	file	uniq(1)
USERFILE: UUCP pathname permissions	file.	USERFILE(5)
access: determine accessibility of a	file.	access(3F)
chmod: change mode of a	file	chmod(3F)
execvp, exec, execve, exect, environ: execute a	file. exect, exect, execte, exectp,	execl(3)
link: make a link to an existing	file.	link(3F)
		· ·
begin logging subroutine calls and close a log	file. VI_Login, VI_Logout:	log(3G)
rename: rename a	file	rename(3F)
VI Run: process a log	file	run(3G)
uuencode: format of an encoded uuencode	file.	uuencode(5)
vipw: edit the password	file.	vipw(8)
versions of object modules were used to construct a	file. what: show what	what(1)
diff: differential	file and directory comparator	diff(1)
bugfiler:	file bug reports in folders automatically.	bugfiler(8)
mkstr: create an error message	file by massaging C source	mkstr(1)
diff3: 3-way differential	file comparison.	diff3(1)
fcntl:	file control.	fcntl(2)
umask: change or display	file creation mask.	csh(1)
umask: set	file creation mode mask	umask(2)
	file: determine file type	file(1)
rdist: remote	file distribution program	rdist(1)
setssent, endssent: get file system descriptor	file entry. /getfsspec, getfsfile, getfstype,	getfsent(3)
		getgrent(3)
getgrgid, getgrnam, setgrent, endgrent: get group	file entry. getgrent,	· · · · ·
setpwent, endpwent, setpwfile: get password	file entry. getpwent, getpwuid, getpwnam,	getpwent(3)
getttynam, setttyent, endttyent: get ttys	file entry. getttyent,	getttyent(3)
grep, egrep, fgrep: search a	file for a pattern	grep(1)
	file for reading or writing, or create a new file	· `
open: open a		open(2)
aliases: aliases	file for sendmail	aliases(5)
rc.config: configuration	file for startup scripts	rc.config(5)
ar: archive (library)	file format	
ar: archive (library)	file format.	ar(5)
tar: tape archive	file format.	tar(5)
tar: tape archive which: locate a program	file format	tar(5) which(1)
tar: tape archive which: locate a program chfn, chsh, passwd: change password	file format.	tar(5)
tar: tape archive which: locate a program	file format	tar(5) which(1)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution	file format	tar(5) which(1) passwd(1) uuxqt(8C)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a	file format. file including aliases and paths (csh only). file information. file interpreter. file into into vidual files. file into pieces. file merger. file name. file on a logical unit.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, stell: reposition a more, page:	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, Istat, fstat: get stat, Istat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pumerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system. file system. file system. file system. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pumerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uwxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, Istat, fstat: get stat, Istat, fstat: get mkfs: construct a mkproto: construct a protype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) newfs(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, Istat, fstat: get stat, Istat, fstat: get stat, Istat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(2) mount(8) newfs(8) repquota(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uwxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, Istat, fstat: get stat, Istat, fstat: get stat, Istat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) repquota(8) setquota(2)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, Istat, fstat: get stat, Istat, fstat: get stat, Istat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(2) mount(8) newfs(8) repquota(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uwxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, Istat, fstat: get stat, Istat, fstat: get stat, Istat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) repquota(8) setquota(2)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck:	file format. file including aliases and paths (csh only). file information. file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get	file format. file including aliases and paths (csh only). file information. file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system consistency check and interactive file system descriptor file entry. /getfsspec,	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck:	file format. file including aliases and paths (csh only). file information. file information. file interpreter. file into pieces. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system consistency check and interactive file system descriptor file entry. /getfsspec, file system directory consistency check.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental	file format. file including aliases and paths (csh only). file information. file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system descriptor file entry. /getfsspec, file system directory consistency check. file system dump.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck:	file format. file including aliases and paths (csh only). file information. file information. file interpreter. file into pieces. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system consistency check and interactive file system descriptor file entry. /getfsspec, file system directory consistency check.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system descriptor file entry. /getfsspec, file system durp. file system dump. file system dump. file system dump.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8) rdump(8C)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uwxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a perege: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental rdump: rrestore: restore a	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system descriptor file entry. /getfsspec, file system descriptor file entry. /getfsspec, file system dump. file system dump. file system dump across the network.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(2) mount(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8) rdump(8C) rrestore(8C)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uwxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, Istat, fstat: get stat, Istat, fstat: get stat, Istat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental rdump: rrestore: restore a dumpfs: dump	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system dispression of the consistency check and interactive file system durp. file system dump across the network. file system information.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8) rdump(8C) rrestore(8C) dumpfs(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental rdump: rrestore: restore a dumpfs: dump quot: summarize	file format. file including aliases and paths (csh only). file information. file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system dispatence or check and interactive file system dump. file system dump across the network. file system dump across the network. file system dump across the network. file system information. file system ownership.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8) rdump(8C) rrestore(8C) dumpfs(8) quot(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental rdump: rrestore: restore a dumpfs: dump quot: summarize quotacheck:	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system dispression of the consistency check and interactive file system durp. file system dump across the network. file system information.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8) rdump(8C) rrestore(8C) dumpfs(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental rdump: rrestore: restore a dumpfs: dump quot: summarize	file format. file including aliases and paths (csh only). file information. file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system dispatence or check and interactive file system dump. file system dump across the network. file system dump across the network. file system dump across the network. file system information. file system ownership.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8) rdump(8C) rrestore(8C) dumpfs(8) quot(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental rdump: rrestore: restore a dumpfs: dump quot: summarize quotacheck: quotaon, quotaoff: turn	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system donsistency check and interactive file system descriptor file entry. /getfsspec, file system dump. file system dump across the network. file system dump across the network. file system ownership. file system quota consistency checker. file system quota consistency checker.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8C) rrestore(8C) dumpfs(8) quotacheck(8) quotacheck(8) quotacheck(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a mkproto: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental rdump: rrestore: restore a dumpfs: dump quot: summarize quotacheck: quotaon, quotaoff: turn restore: incremental	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system descriptor file entry. /getfsspec, file system dump across the network. file system dump across the network. file system ownership. file system quota consistency checker. file system quota consistency checker. file system quotas on and off. file system restore.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8C) rrestore(8C) dumpfs(8) quot(8) quotacheck(8) quotaon(8) restore(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental rdump: rrestore: restore a dumpfs: dump quot: summarize quotacheck: quotaon, quotaoff: turn restore: incremental restore: incremental restore: incremental	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system descriptor file entry. /getfsspec, file system dump across the network. file system dump across the network. file system dump across the network. file system ownership. file system quota consistency checker. file system quotas on and off. file system restore.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8C) rrestore(8C) dumpfs(8) quot(8) quotacheck(8) quotacheck(8) restore(8) restore(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental rdump: rrestore: restore a dumpfs: dump quot: summarize quotacheck: quotaon, quotaoff: turn restore: incremental restore: incremental icheck:	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system descriptor file entry. /getfsspec, file system dump. file system dump across the network. file system dump across the network. file system information. file system quota consistency checker. file system quota on and off. file system restore. file system restore. file system restore. file system restore.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8) rdump(8C) rrestore(8C) dumpfs(8) quot(8) quotacheck(8) restore(8) restore(8) restore(8)
tar: tape archive which: locate a program chfn, chsh, passwd: change password uuxqt: UUCP execution fsplit: split a multi-routine Fortran split: split a pmerge: pascal mktemp: make a unique fseek, ftell: reposition a more, page: stat, lstat, fstat: get stat, lstat, fstat: get stat, lstat, fstat: get mkfs: construct a prototype mount, umount: mount or remove mount, umount: mount and dismount newfs: construct a new newfs: construct a new repquota: summarize quotas for a setquota: enable/disable quotas on a tunefs: tune up an existing repair. fsck: getfsfile, getfstype, setfsent, endfsent: get dcheck: dump: incremental rdump: rrestore: restore a dumpfs: dump quot: summarize quotacheck: quotaon, quotaoff: turn restore: incremental restore: incremental restore: incremental	file format. file including aliases and paths (csh only). file information. file interpreter. file into individual files. file into pieces. file merger. file name. file on a logical unit. file perusal filter for crt viewing. file status. file status. file system. file system descriptor file entry. /getfsspec, file system dump across the network. file system dump across the network. file system dump across the network. file system ownership. file system quota consistency checker. file system quotas on and off. file system restore.	tar(5) which(1) passwd(1) uuxqt(8C) fsplit(1) split(1) pmerge(1) mktemp(3) fseek(3F) more(1) stat(2) stat(3F) mkfs(8) mkproto(8) mount(2) mount(8) newfs(8) repquota(8) setquota(2) tunefs(8) fsck(8) getfsent(3) dcheck(8) dump(8C) rrestore(8C) dumpfs(8) quot(8) quotacheck(8) quotacheck(8) restore(8) restore(8)

fs, inode: format of	C1	C-/C)
utime: set	file system volume	fs(5) utime(3C)
utimes: set	file times.	utimes(2)
truncate, ftruncate: truncate a	file to a specified length.	truncate(2)
ftpd: DARPA Internet	File Transfer Protocol server	ftpd(8C)
tftpd: DARPA Trivial	File Transfer Protocol server	tftpd(8Ć)
file: determine	file type	file(1)
basename: strip	filename affixes	basename(1)
glob:	filename expand argument list	csh(1)
ferror, feof, clearerr,	fileno: stream status inquiries	ferror(3S)
checknr: check nroff/troff	files	checknr(1)
cmp: compare two	files.	cmp(1)
comm: select or reject lines common to two sorted	files.	comm(1)
config: build system configuration config: build system configuration	files	config(8)
find: find	files.	config(8) find(1)
split a multi-routine Fortran file into individual	files. fsplit:	fsplit(1)
makedev: make system special	files.	makedev(8)
makedev: make system special	files.	makedev(8)
mv: move or rename	files.	mv(1)
omerge: merge object	files	omerge(8)
rmdir, rm: remove (unlink) directories or	files	rmdir(1)
sort: sort or merge	files	sort(1)
intro: introduction to special	files and hardware support.	intro(4)
intro: introduction to special	files and hardware support.	intro(4)
catman: create the cat	files for the manual.	catman(8)
fsync: synchronize a	file's in-core state with that on disk	fsync(2)
pprint: print text ptroff: print troff	files on IBM 3812 Pageprinter	pprint(1)
rm, rmdir: remove (unlink)	files on IBM 3812 Pageprinter	ptroff(1) rm(1)
uucico, uucpd: transfer	files queued by uucp or uux.	uucico(8C)
badsect: create	files to contain bad sectors.	badsect(8)
badsect: create	files to contain bad sectors.	badsect(8)
newvd: create a new	filesystem on a Remote Virtual Disk (RVD)	newvd(8)
fstab: static information about the	filesystems.	fstab(5)
5152 Graphics Printer nroff post-processing	filter. prfl: IBM 4201 Proprinter/IBM	prfl(1)
more, page: file perusal	filter for crt viewing	more(1)
colpro: column	filter for IBM 4201 Proprinter	colpro(1)
ppt: spooling system	filter for the IBM 3812 Pageprinter	ppt(8)
colcrt:	filter nroff output for CRT previewing	colcrt(1)
col:	filter reverse line feeds	col(1)
plot: graphics	filters	plot(1G)
Graphics Printer. ibmbit, ibmgra, ibmpro: output refer:	filters for the IBM 4201 Proprinter and IBM 5152 find and insert literature references in documents	lpfilter(8r)
find:	find and insert literature references in documents find files	refer(1) find(1)
illia.	find: find files.	find(1)
look:	find lines in a sorted list.	look(1)
manual. man:		
	find manual information by keywords; print out the .	man(1)
ttyname, isatty, ttyslot:	find manual information by keywords; print out the find name of a terminal	, , ,
ttyname, isatty, ttyslot:	find name of a terminal	man(1)
ttyname, isatty, ttyslot: ttynam, isatty: lorder:	find name of a terminal	man(1) ttyname(3) ttynam(3F) lorder(1)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography,	find name of a terminal	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout:	find name of a terminal	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography,	find name of a terminal	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout:	find name of a terminal	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings:	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server.	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered,	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb,	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem,	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server.	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) fingerd(8C) ieee(3)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered,	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) fingerd(8C) ieee(3) ieee(3M)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries. first few lines.	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) fingerd(8C) ieee(3) ieee(3M) fold(1)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete,	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries. first few lines. firstkey, nextkey: data base subroutines.	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) finger(8C) ieee(3) ieee(3M) fold(1) plot(3F)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries first few lines. firstkey, nextkey: data base subroutines. (sh only).	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) finger(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority ptfinstall: install a Program Temporary	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries first few lines. firstkey, nextkey: data base subroutines. (sh only). Fix (PTF).	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) finger(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries first few lines. first key, nextkey: data base subroutines. (sh only). Fix (PTF). xmh: X window interface to the mh Mail	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) finger(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8) xmh(1)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority ptfinstall: install a Program Temporary Handler.	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries first few lines. first few lines. first key, nextkey: data base subroutines. (sh only). Fix (PTF). xmh: X window interface to the mh Mail fl: console floppy interface.	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) finger(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8) xmh(1) fl(4)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority ptfinstall: install a Program Temporary	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries first few lines. firstkey, nextkey: data base subroutines. (sh only). Fix (PTF). xmh: X window interface to the mh Mail fl: console floppy interface. floopy: archiver and copier for floppy.	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) fingerd(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8) xmh(1) fl(4) arff(8V)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority ptfinstall: install a Program Temporary Handler. arff,	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries. first few lines. first few lines. firstkey, nextkey: data base subroutines. (sh only). Fix (PTF). xmh: X window interface to the mh Mail fl: console floppy interface. flcopy: archiver and copier for floppy. flcopy: copier for diskettes.	man(1) ttyname(3) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) fingerd(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8) xmh(1) fl(4) arff(8V) flcopy(8r)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority ptfinstall: install a Program Temporary Handler. arff, extreme values. flmin,	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries first few lines. first few lines. firstkey, nextkey: data base subroutines. (sh only). Fix (PTF). xmh: X window interface to the mh Mail fl: console floppy interface. flcopy: archiver and copier for floppy. flcopy: copier for diskettes. flmax, ffrac, dflmin, dflmax, dffrac, inmax: return	man(1) ttyname(3) ttyname(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) fingerd(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8) xmh(1) fi(4) arff(8V) ficopy(8r) fimin(3F)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority ptfinstall: install a Program Temporary Handler. arff, extreme values. fimin, return extreme values.	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries. first few lines. first few lines. firstkey, nextkey: data base subroutines. (sh only). Fix (PTF). xmh: X window interface to the mh Mail fl: console floppy interface. flcopy: archiver and copier for floppy. flcopy: copier for diskettes.	man(1) ttyname(3) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) fingerd(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8) xmh(1) fl(4) arff(8V) flcopy(8r)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority ptfinstall: install a Program Temporary Handler. arff, extreme values. fimin, return extreme values. afpacode: load, test, and bring online the Advanced	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries first few lines. first few lines. firstkey, nextkey: data base subroutines. (sh only). Fix (PTF). xmh: X window interface to the mh Mail fl: console floppy interface. flcopy: archiver and copier for floppy. flcopy: copier for diskettes. flmax, ffrac, dflmin, dflmax, dffrac, inmax: return flmin, flmax, ffrac, dflmin, dflmax, dffrac, inmax: Floating Point Accelerator.	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) finger(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8) xmh(1) fl(4) arff(8V) flcopy(8r) flmin(3F)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority ptfinstall: install a Program Temporary Handler. arff, extreme values. fimin, return extreme values.	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries first few lines. first few lines. firstkey, nextkey: data base subroutines. (sh only). Fix (PTF). xmh: X window interface to the mh Mail fl: console floppy interface. flcopy: archiver and copier for floppy. flcopy: copier for diskettes. flmax, ffrac, dflmin, dflmax, dffrac, inmax: return flmin, flmax, ffrac, dflmin, dflmax, dffrac, inmax:	man(1) ttyname(3) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) finger(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8) xmh(1) fl(4) arfl(8V) flcopy(8r) flmin(3F) flmin(3F) afpacode(8r)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority ptfinstall: install a Program Temporary Handler. arff, extreme values. flmin, return extreme values. afpacode: load, test, and bring online the Advanced fpa: direct interface to trpfpe, fpecnt: trap and repair	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries first few lines. first key, nextkey: data base subroutines. (sh only). Fix (PTF). xmh: X window interface to the mh Mail fl: console floppy interface. flcopy: archiver and copier for floppy. flcopy: copier for diskettes. flmax, ffrac, dflmin, dflmax, dffrac, inmax: return flmin, flmax, ffrac, dflmin, dflmax, dffrac, inmax: Floating Point Accelerator. floating point faults. floating point overflow.	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) fingerd(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8) xmh(1) fl(4) arff(8V) flcopy(8r) flmin(3F) afpacode(8r) fpa(3X) trpfpe(3F) trapov(3F)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority ptfinstall: install a Program Temporary Handler. arff, extreme values. flmin, return extreme values. afpacode: load, test, and bring online the Advanced fpa: direct interface to trpfpe, fpecnt: trap and repair trapov: trap and repair getfloatstate: return machine and process	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries. first few lines. first few lines. firstkey, nextkey: data base subroutines. (sh only). Fix (PTF). xmh: X window interface to the mh Mail fi: console floppy interface. floopy: archiver and copier for floppy. floopy: copier for diskettes. flmax, ffrac, dflmin, dflmax, dffrac, inmax: return flmin, flmax, ffrac, dflmin, dflmax, dffrac, inmax: Floating Point Accelerator. floating point accelerator. floating point faults. floating point overflow. floating point state.	man(1) ttyname(3) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) fingerd(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8) xmh(1) ff(4) arff(8V) flcopy(8r) flmin(3F) flmin(3F) fapacode(8r) fpa(3X) trpfpe(3F) trapov(3F) getfloatstate(2)
ttyname, isatty, ttyslot: ttynam, isatty: lorder: lookbib: build inverted index for a bibliography, spell, spellin, spellout: binary, file. strings: rint, classdouble, classfloat, isnan, unordered, manipulations. copysign, drem, fold: fold long lines for plot: openpl et al.: f77 library interface to head: give dbminit, fetch, store, delete, nice, nohup: run a command at low priority ptfinstall: install a Program Temporary Handler. arff, extreme values. flmin, return extreme values. afpacode: load, test, and bring online the Advanced fpa: direct interface to trpfpe, fpecnt: trap and repair	find name of a terminal. find name of a terminal port. find ordering relation for an object library. find references in a bibliography. indxbib, find spelling errors. find the printable strings in a object, or other finger: user information lookup program. fingerd: remote user information server. finite, infinity, nextdouble, nextfloat,//scalb, finite, logb, scalb: copysign, remainder, exponent finite width output device. plot (3X) libraries first few lines. first key, nextkey: data base subroutines. (sh only). Fix (PTF). xmh: X window interface to the mh Mail fl: console floppy interface. flcopy: archiver and copier for floppy. flcopy: copier for diskettes. flmax, ffrac, dflmin, dflmax, dffrac, inmax: return flmin, flmax, ffrac, dflmin, dflmax, dffrac, inmax: Floating Point Accelerator. floating point faults. floating point overflow.	man(1) ttyname(3) ttynam(3F) lorder(1) lookbib(1) spell(1) strings(1) finger(1) fingerd(8C) ieee(3) ieee(3M) fold(1) plot(3F) head(1) dbm(3X) nice(1) ptfinstall(8) xmh(1) fl(4) arff(8V) flcopy(8r) flmin(3F) afpacode(8r) fpa(3X) trpfpe(3F) trapov(3F)

infnan: signals invalid	floating-point operations on a VAX (temporary)	infnan(3M)
file.	flock: apply or remove an advisory lock on an open	flock(2)
functions. fabs,		
fabs, floor, ceil: absolute value,		floor(3M)
	floor, ceiling functions.	floor(3M)
arff, flcopy: archiver and copier for	floppy.	arff(8V)
rx: DEC RX02	floppy disk interface	rx(4)
rxformat: format	floppy disks.	rxformat(8V)
fi: console	floppy interface	fl(4)
fclose, fflush: close or	flush a stream.	fclose(3S)
	flush: flush output to a logical unit.	flush(3F)
flush:	flush output to a logical unit.	flush(3F)
exit: terminate a process after	flushing any pending output.	exit(3)
/gcd, invert, rpow, msqrt, mcmp, move, min, omin,	fmin, m_in, mout, omout, fmout, m_out, sdiv, itom:/	mp(3X)
/mcmp, move, min, omin, fmin, m_in, mout, omout,	fmout, m_out, sdiv, itom: multiple precision/	mp(3X)
_ · · · ·	fmt: simple text formatter	fmt(1)
device.	fold: fold long lines for finite width output	fold(1)
fold:	fold long lines for finite width output device	fold(1)
bugfiler: file bug reports in	folders automatically	bugfiler(8)
vwidth: make troff width table for a	font.	vwidth(1)
xfd - X window system	font displayer.	xfd(1)
vfont:	font formats for the Benson-Varian or Versatec	vfont(5)
xlsfonts - X window system		
•		xlsfonts(1)
font3812:	font structures for 3812 fonts	font3812(5)
C .0010 C	font3812: font structures for 3812 fonts	font3812(5)
font3812: font structures for 3812	fonts.	font3812(5)
VI_GetFont, VI_DropFont: select and manipulate	fonts. VI_Font,	font(3G)
build width tables for IBM 3812 Pageprinter	fonts. width3812:	width3812(8)
/cvt20to12, cvt00to12: convert IBM 3820 and IBM 3800	fonts for use with the IBM 3812 Pageprinter	cvt3812(8)
	fopen, freopen, fdopen: open a stream	fopen(3S)
VI_Force:	force output of graphics orders	force(3G)
rvdshut:	force shutdown of Remote Virtual Disk (RVD) server	rvdshut(8)
rvddown:	force spindown of a Remote Virtual Disk (RVD) pack	rvddown(8)
	foreach: loop over list of names	csh(1)
fg: bring job into	foreground	csh(1)
8 6 1	fork: create a copy of this process	fork(3F)
	fork: create a new process.	fork(2)
idate, itime: return date or time in numerical	form.	idate(3F)
dmesg: collect system diagnostic messages to	form error log.	dmesg(8)
· · · · · · · · · · · · · · · · · · ·	format.	
ar: archive (library) file		ar(5)
dump, dumpdates: incremental dump	format.	dump(5)
tar: tape archive file	format.	tar(5)
indent: indent and	format C program source	indent(1)
format: how to	format disk packs	format(8V)
fdformat:	format diskettes	fdformat(8r)
rxformat:	format floppy disks	rxformat(8V)
	format: format hard disks	format(8r)
format:	format hard disks	format(8r)
htable: convert NIC standard	format host tables	htable(8)
gettable: get NIC	format host tables from a host	gettable(8C)
	format: how to format disk packs	format(8V)
vtroff, or troff. vlp:	Format Lisp programs to be printed with nroff,	vlp(1)
uuencode:	format of an encoded unencode file.	uuencode(5)
dir:		dir(5)
fs, inode:	format of file system volume.	fs(5)
core:	format of memory image file.	core(5)
core:	format of memory image file.	core(5)
disk:	format of reserved areas of the hard disk	disk(4)
tbl:	format tables for nroff or troff	tbl(1)
scsiformat:	format the IBM 9332 disk unit.	scsiformat(8c)
	_	` '
tp: DEC/mag tape	formats.	tp(5)
vfont: font	formats for the Benson-Varian or Versatec	vfont(5)
scanf, fscanf, sscanf:	formatted input conversion	scanf(3S)
printf, fprintf, sprintf:	formatted output conversion.	printf(3S)
fmt: simple text	formatter	fmt(1)
nroff: text	formatting.	nroff(1)
troff, nroff: text	formatting and typesetting.	troff(1)
f77:	Fortran 77 compiler	f77(1)
f77:	FORTRAN 77 compiler	f77(1)
ratfor: rational	Fortran dialect	ratfor(1)
fpr: print	Fortran file	fpr(1)
fsplit: split a multi-routine	Fortran file into individual files	fsplit(1)
efl: Extended	Fortran Language	efi(1)
intro: introduction to	FORTRAN library functions	intro(3F)
putc, fputc: write a character to a	fortran logical unit.	putc(3F)
struct: structure	Fortran programs.	struct(1)
login,/ sh, for, case, if, while, :, .	, break, continue, cd, eval, exec, exit, export,	sh(1)
Q , , , , , , , , , , , , , , , , , , ,	, , ,,,,, error,	\" <i>,</i>

		4 / 4 3
exit, export,/ sh, for, case, if, while,:	, ., break, continue, cd, eval, exec,	
compiler/interpreter.	fp: Functional Programming language	
accelerator.	fpa: direct interface to floating point	
trpfpe,	fpecnt: trap and repair floating point faults	trpfpe(3F)
• • •	fpr: print Fortran file	fpr(1)
printf,	fprintf, sprintf: formatted output conversion	
fptestround, fpsetround, swapround, fptestflag,	fpsetflag, /infinity, nextdouble, nextfloat,	
/infinity, nextdouble, nextfloat, fptestround,	fpsetround, swapround, fptestflag, fpsetflag,	
nextfloat, fptestround, fpsetround, swapround,	fptestflag, fpsetflag, /infinity, nextdouble,	
/unordered, finite, infinity, nextdouble, nextfloat,	sptestround, swapround, sptestslag,/	ieee(3)
putc, putchar,	fputc, putw: put character or word on a stream	
putc,	fputc: write a character to a fortran logical unit	putc(3F)
puts,	fputs: put a string on a stream.	puts(3S)
ik: Ikonas	frame buffer, graphics device interface	ik(4)
liszt: compile a	Franz Lisp program	
non vonpus	fread, fwrite: buffered binary input/output	
df: disk		
	free	
malloc,	free, falloc: memory allocator.	
malloc,	free, realloc, calloc, alloca: memory allocator	
fopen,	freopen, fdopen: open a stream.	fopen(3S)
exponent.	frexp, Idexp, modf: split into mantissa and	frexp(3)
from: who is my mail	from?	from(1)
sccs:	front end for the SCCS subsystem.	sccs(1)
scs.	fs, inode: format of file system volume	
. •		
scanf,	fscanf, sscanf: formatted input conversion	
mklost + found: make a lost + found directory for	fsck	
repair.	fsck: file system consistency check and interactive	
	fseek, ftell: reposition a file on a logical unit	fseek(3F)
· ·	seek, stell, rewind: reposition a stream	ſseek(3S)
individual files.	fsplit: split a multi-routine Fortran file into	fsplit(1)
	fstab: static information about the filesystems	fstab(5)
etat letat	fstat: get file status.	
stat, Istat,		stat(2)
stat, lstat,	fstat: get file status.	stat(3F)
on disk.	fsync: synchronize a file's in-core state with that	fsync(2)
fseek,	ftell: reposition a file on a logical unit	fseek(3F)
fseek,	ftell, rewind: reposition a stream	fseek(3S)
time,	ftime: get date and time	time(3C)
	ftpd: DARPA Internet File Transfer Protocol server	
truncate,	ftruncate: truncate a file to a specified length	
shutdown: shut down part of a	full-duplex connection.	shutdown(2)
		011 at a 0 11 11 (2)
•		tn3270(1)
tn3270:	full-screen remote login to IBM VM/CMS	
tn3270: tn3270:	full-screen remote login to IBM VM/CMS full-screen remote login to IBM VM/CMS	tn3270(1)
tn3270: tn3270: lgamma: log gamma	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function.	tn3270(1) lgamma(3M)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp:	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language	tn3270(1) lgamma(3M) fp(1)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions.	tn3270(1) lgamma(3M) fp(1) asinh(3M)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions. functions.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions. functions.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions. functions. functions. functions. functions.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions. functions. functions. functions. functions. functions. functions.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) etf(3M) floor(3M) intro(3) intro(3F)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses. sin, functions: of two kinds for integer orders.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sinh(3M) bessel(3F)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses. sin, functions: of two kinds for integer orders. functions with "optimal" cursor motion.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sinh(3M) bessel(3F) curses(3X)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses. sin, functions: of two kinds for integer orders.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sinh(3M) bessel(3F)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread,	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses. sin, functions: of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses. sin, functions: of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread,	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses. sin, functions sin functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin,	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sinh(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow,	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses. sin, functions of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. ged, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, Ishift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow,	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses. sin, functions of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. govt: output conversion.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3)
tn3270: tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon:	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions functions. functions. functions. functions. functions. functions and their inverses. sin, functions: of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort:	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses. sin, functions: of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey:	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses sin, functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts:	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses. sin, functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key. generate hashed host table.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8) mkhosts(8)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey:	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions functions functions functions functions functions and their inverses. sin, functions of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. ged, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gevt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key. generate hashed host table. generate hashed password table.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts:	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions and their inverses. sin, functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key. generate hashed host table.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8) mkhosts(8)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts: mkpasswd:	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions functions functions functions functions functions and their inverses. sin, functions of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. ged, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gevt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key. generate hashed host table. generate hashed password table.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecv(13) kgmon(8) abort(3) makekey(8) mkhosts(8)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts: mkpasswd: lptest: ncheck:	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key. generate hashed password table. generate lineprinter ripple pattern. generate names from i-numbers.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8) mkhosts(8) mkpasswd(8) lptest(1) ncheck(8)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts: mkpasswd: lptest: ncheck: rand, srand: random number	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions functions functions functions functions control functions of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key. generate hashed host table. generate lineprinter ripple pattern. generate names from i-numbers. generator.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8) mkpasswd(8) lptest(1) ncheck(8) rand(3C)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts: mkpasswd: lptest: ncheck: rand, srand: random number random, drandm, irandm: better random number	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key. generate hashed host table. generate lineprinter ripple pattern. generate names from i-numbers. generator.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sinh(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8) mkhosts(8) mkpasswd(8) lptest(1) ncheck(8) rand(3C) random(3F)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts: mkpasswd: lptest: ncheck: rand, srand: random number random, drandm, irandm: better random number lex:	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions. functions. functions. functions. functions. functions. functions. functions and their inverses. sin, functions sit wo kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key. generate hashed host table. generate lineprinter ripple pattern. generator. generator. generator. generator of lexical analysis programs.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8) mkhosts(8) mkpasswd(8) lptest(1) ncheck(8) rand(3C) random(3F) lex(1)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts: mkpasswd: lptest: ncheck: rand, srand: random number random, drandm, irandm: better random number lex: /srandom, initstate, setstate: better random number	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions functions functions functions functions functions functions functions functions and their inverses sin, functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key. generate hashed host table. generate lineprinter ripple pattern. generate names from i-numbers. generator. generator. generator; routines for changing generators.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8) mkhosts(8) mkpasswd(8) lptest(1) ncheck(8) rand(3C) random(3F) lex(1) random(3)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts: mkpasswd: lptest: ncheck: rand, srand: random number random, drandm, irandm: better random number random number generator; routines for changing	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions functions functions functions functions functions functions functions and their inverses. sin, functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate a fault. generate a fault. generate names from i-numbers. generate names from i-numbers. generator. generator; routines for changing generators. generators. /srandom, initstate, setstate: better	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8) mkhosts(8) mkpasswd(8) lptest(1) ncheck(8) rand(3C) random(3F) lex(1) random(3) random(3)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts: mkpasswd: lptest: ncheck: rand, srand: random number random, drandm, irandm: better random number lex: /srandom, initstate, setstate: better random number	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions functions functions functions functions and their inverses sin, functions of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key. generate hashed host table. generate lineprinter ripple pattern. generate names from i-numbers. generator. generator. generator; routines for changing generators. generators. /srandom, initstate, setstate: better gerror, ierrno: get system error messages.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) ffloor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8) mkpasswd(8) lptest(1) ncheck(8) random(3F) lex(1) random(3) random(3) perror(3F)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts: mkpasswd: lptest: ncheck: rand, srand: random number random, drandm, irandm: better random number random number generator; routines for changing	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions functions functions functions functions functions of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate caryption key. generate hashed password table. generate lineprinter ripple pattern. generate names from i-numbers. generator. generator. generator; routines for changing generators. generators. /srandom, initstate, setstate: better gerror, ierrno: get system error messages. getarg, iarge: return command line arguments.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8) mkhosts(8) mkpasswd(8) lptest(1) ncheck(8) rand(3C) random(3F) lex(1) random(3) random(3) perror(3F) getarg(3F)
tn3270: tn3270: lgamma: log gamma compiler/interpreter. fp: asinh, acosh, atanh: inverse hyperbolic bit: and, or, xor, not, rshift, lshift bitwise erf, erfc: error fabs, floor, ceil: absolute value, floor, ceiling intro: introduction to C library intro: introduction to FORTRAN library j0, j1, jn, y0, y1, yn: Bessel math: introduction to mathematical library sinh, cosh, tanh: hyperbolic cos, tan, asin, acos, atan, atan2: trigonometric bessel curses: screen fread, lgamma: log fmin, m_in, mout,/ madd, msub, mult, mdiv, pow, ecvt, fcvt, buffers. kgmon: abort: makekey: mkhosts: mkpasswd: lptest: ncheck: rand, srand: random number random, drandm, irandm: better random number random number generator; routines for changing	full-screen remote login to IBM VM/CMS. full-screen remote login to IBM VM/CMS. function. Functional Programming language functions. functions functions functions functions functions and their inverses sin, functions of two kinds for integer orders. functions with "optimal" cursor motion. fwrite: buffered binary input/output. gamma function. gcd, invert, rpow, msqrt, mcmp, move, min, omin, gcore: get core images of running processes. gcvt: output conversion. generate a dump of the operating system's profile generate a fault. generate encryption key. generate hashed host table. generate lineprinter ripple pattern. generate names from i-numbers. generator. generator. generator; routines for changing generators. generators. /srandom, initstate, setstate: better gerror, ierrno: get system error messages.	tn3270(1) lgamma(3M) fp(1) asinh(3M) bit(3F) erf(3M) floor(3M) intro(3) intro(3F) j0(3M) math(3M) sinh(3M) sin(3M) bessel(3F) curses(3X) fread(3S) lgamma(3M) mp(3X) gcore(1) ecvt(3) kgmon(8) abort(3) makekey(8) mkhosts(8) mkpasswd(8) lptest(1) ncheck(8) rand(3C) random(3F) lex(1) random(3) random(3) perror(3F) getarg(3F)

from stream.	getc, getchar, fgetc, getw: get character or word	getc(3S)
stream. getc,	getchar, fgetc, getw: get character or word from	getc(3S)
	getcwd: get pathname of current working directory	getcwd(3F)
	getdiskbyname: get disk description by its name	getdisk(3)
	getdtablesize: get descriptor table size	getdtablesize(2)
getgid,	getegid: get group identity.	getgid(2)
geigiu,	getegra, get group rachity.	getgiu(z)
	getenv: get value of environment variables	getenv(3F)
	getenv: value for environment name	getenv(3)
getuid,	geteuid: get user identity.	getuid(2)
point state.	getfloatstate: return machine and process floating	getfloatstate(2)
emulator.	getspemulator: return address of the floating-point	getfpemulator(2)
setfsent, endfsent: get file system descriptor/	getssent, getsspec, getssile, getstype,	getsent(3)
system descriptor file entry. getfsent, getfsspec,	getssfile, getsstype, setssent, endssent: get file	getfsent(3)
endfsent: get file system descriptor/ getfsent,	getfsspec, getfsfile, getfstype, setfsent,	getfsent(3)
	gettsspee, gettstile, gettstype, settsetit,	getisent(3)
descriptor file/ getfsent, getfsspec, getfsfile,	getsstype, setsent, endssent: get file system	getfsent(3)
getuid,	getgid: get user or group ID of the caller	getuid(3F)
	getgid, getegid: get group identity.	getgid(2)
get group file entry.	getgrent, getgrgid, getgrnam, setgrent, endgrent:	getgrent(3)
file entry. getgrent,	getgrgid, getgrnam, setgrent, endgrent: get group	getgrent(3)
getgrent, getgrgid,	getgrnam, setgrent, endgrent: get group file entry	
8-8, 8-8-8,	getgroups: get group access list	
got network host entry gothosthynama		
get network host entry. gethostbyname,	gethostbyaddr, gethostent, sethostent, endhostent:	
sethostent, endhostent: get network host entry.	gethostbyname, gethostbyaddr, gethostent,	
host entry. gethostbyname, gethostbyaddr,	gethostent, sethostent, endhostent: get network	
current host.	gethostid, sethostid: get/set unique identifier of	gethostid(2)
host.	gethostname, sethostname: get/set name of current	gethostname(2)
timer.	getitimer, setitimer: get/set value of interval	
VIII.VII		
	getlog: get user's login name	getlog(3F)
	getlogin: get login name.	
get network entry. getnetent,	getnetbyaddr, getnetbyname, setnetent, endnetent:	
entry. getnetent, getnetbyaddr,	getnetbyname, setnetent, endnetent; get network	
endnetent: get network entry.	getnetent, getnetbyaddr, getnetbyname, setnetent,	getnetent(3N)
·	getopt: get option letter from argv	getopt(3)
	getpagesize: get system page size.	getpagesize(2)
	gotpages road a naceword	getpagesize(2)
	getpass: read a password.	getpass(3)
	getpeername: get name of connected peer	getpeername(2)
	getpgrp: get process group	getpgrp(2)
	getpid: get process id	getpid(3F)
	getpid, getppid: get process identification	getpid(2)
getpid,	getppid: get process identification	getpid(2)
scheduling priority.	getpriority, setpriority: get/set program	getpriority(2)
protocol entry. getprotoent, getprotobynumber,	getprotobyname, setprotoent, endprotoent: get	
endprotoent: get protocol entry. getprotoent,	getprotobynumber, getprotobyname, setprotoent,	
setprotoent, endprotoent: get protocol entry.	getprotoent, getprotobynumber, getprotobyname,	getprotoent(3N)
	getpw: get name from uid	getpw(3C)
setpwfile: get password file entry.	getpwent, getpwuid, getpwnam, setpwent, endpwent,	
password file entry. getpwent, getpwuid,	getpwnam, setpwent, endpwent, setpwfile: get	- : :-:
get password file entry. getpwent,	getpwuid, getpwnam, setpwent, endpwent, setpwfile: .	getpwent(3)
	cotribuit catelimit control maximum evetem	setrlimit(2)
resource consumption.	getrlimit, setrlimit: control maximum system	
utilization.	getrusage: get information about resource	
	gets, fgets: get a string from a stream	gets(3S)
entry. getservent, getservbyport,	getservbyname, setservent, endservent: get service	getservent(3N)
endservent: get service entry. getservent,	getservbyport, getservbyname, setservent,	
setservent, endservent: get service entry.	getservent, getservbyport, getservbyname,	getservent(3N)
gettimeofday, settimeofday:	get/set date and time	
gethostname, sethostname:	get/set name of current host.	
getpriority, setpriority:	get/set program scheduling priority.	
gethostid, sethostid:	get/set unique identifier of current host	
getitimer, setitimer:	get/set value of interval timer	
	getsockname: get socket name.	getsockname(2)
sockets.	getsockopt, setsockopt: get and set options on	getsockopt(2)
	gettable: get NIC format host tables from a host	gettable(8C)
	gettimeofday, settimeofday: get/set date and time	
ttys file entry.	gettlyent, gettlynam, settlyent, endttyent; get	gettimeoiday(2)
		• • • •
entry. getityent,	getttynam, setttyent, endttyent: get ttys file	getttyent(3)
	getty: set terminal mode.	• • • • • • • • • • • • • • • • • • • •
	gettytab: terminal configuration data base	gettytab(5)
	getuid, geteuid: get user identity	
	getuid, getgid: get user or group ID of the caller	
user shells.	getusershell, setusershell, endusershell: get legal	
getc, getchar, fgetc,	getw: get character or word from stream	
	getwd: get current working directory pathname	
head:	give first few lines	head(1)
shutdown: close down the system at a		
	give first few lines	head(1) shutdown(8)
	give first few lines	head(1) shutdown(8) csh(1)

time, ctime, Itime,	gmtime: return system time.	time(3F)
setjmp, longjmp: non-local	goto.	
, I, D	goto: command transfer	
	gprof: display call graph profile data	
graph: draw a		0 1 1 /
gprof: display call	graph: draw a graph	
aedrunner: execute	graphics commands in a log file.	gprof(1) aedrunner(1)
ik: Ikonas frame buffer,	graphics device interface.	
ps: Evans and Sutherland Picture System 2	graphics device interface	
ibm5154, ega: IBM 5154 Enhanced	Graphics Display interface	ibm5154(4)
ibm6153, apa8: IBM 6153 Advanced Monochrome	Graphics Display interface	
ibm6154, apa8c: IBM 6154 Advanced Color	Graphics Display interface	
ibm6155, apa16: IBM 6155 Extended Monochrome plot:	Graphics Display interface	ibm6155(4) plot(1G)
plot:	graphics interface.	
arc, move, cont, point, linemod, space, closepl:	graphics interface. /erase, label, line, circle,	
Information Systems experimental display. aedemul:	graphics interfaces for the IBM Academic	
VI_Force: force output of	graphics orders.	
VI_QFont, VI_QMerge, VI_QPoint, VI_QWidth: query	graphics parameters. /VI_QColor, VI_QDash,	query(3G)
filters for the IBM 4201 Proprinter and IBM 5152 nroff for the IBM 4201 Proprinter and IBM 5152	Graphics Printer. ibmbit, ibmgra, ibmpro: output Graphics Printer. proff:	
prfl: IBM 4201 Proprinter/IBM 5152	Graphics Printer profit post-processing filter	
intro: introduction to display	graphics subroutines.	
lib2648: subroutines for the HP 2648	graphics terminal	
	grep, egrep, fgrep: search a file for a pattern	
vgrind:	grind nice listings of programs.	
Pageprinter. vgrind:	grind nice listings of programs for the IBM 3812	vgrind(1)
chgrp: change getpgrp: get process	group.	
killpg: send signal to a process	group.	
setpgrp; set process	group.	setpgrp(2)
getgroups: get	group access list	
setgroups: set	group access list	setgroups(2)
initgroups: initialize	group access list.	0 1 1 7
group:	group file	group(5)
gergigia, gerginam, sergient, enagient. ger	group: group file.	getgrent(3) group(5)
setregid: set real and effective	group ID.	
setruid, setgid, setegid, setrgid: set user and	group ID. setuid, seteuid,	
getuid, getgid: get user or	group ID of the caller	
getgid, getegid: get	group identity.	
groups: show chown, fchown: change owner and	group memberships	groups(1) chown(2)
make: maintain program	groups	make(1)
	groups: show group memberships	
stty,		stty(3C)
stop:	halt a job or process	
reboot: reboot system or	halt processor.	reboot(2)
	halt: stop the processor. halt: stop the processor.	halt(8) halt(8)
rmail:	handle remote mail received via uucp.	rmail(1)
re_comp, re_exec: regular expression	handler.	1
xmh: X window interface to the mh Mail	Handler	xmh(1)
vhangup: virtually	"hangup" the current control terminal	vhangup(2)
nohup: run command immune to crash: what	hangups	csh(1)
crash: what	happens when the system crashes	crash(8r) crash(8V)
disk: format of reserved areas of the	hard disk	disk(4)
hd:	hard disk interface	
format: format	hard disks.	format(8r)
link: make a	hard link to a file.	` '
support: intro: introduction to special files and	hardware and software support information	support(1) intro(4)
intro: introduction to special files and	hardware support	intro(4)
rehash: recompute command	hash table.	csh(1)
unhash: discard command	hash table	csh(1)
mkhosts: generate	hashed host table	mkhosts(8)
mkpasswd: generate	hashed password table	mkpasswd(8)
hashstat: print command	hashing statistics	csh(1)
leave: remind you when you	hashstat: print command hashing statistics	csh(1) leave(1)
icave. Tenning you when you	hc: High C compiler	hc(1)
	hd: hard disk interface	
	hdh: ACC IF-11/HDH IMP interface	hdh(4)
od: octal, decimal,	hex, ascii dump.	od(1)

t	High Committee	h-/1)
nc: history: print	High C compiler	hc(1) csh(1)
mstory. print	history: print history event list.	csh(1)
	hk: RK6-11/RK06 and RK07 moving head disk	hk(4)
sethostid: get/set unique identifier of current	host. gethostid,	gethostid(2)
gethostname, sethostname: get/set name of current	host.	gethostname(2)
gettable: get NIC format host tables from a	host.	gettable(8C)
hostnm: get name of current htonl, htons, ntohl, ntohs: convert values between	host	hostnm(3F) byteorder(3N)
L.sys: UUCP remote	host description file.	L.sys(5)
remote: remote	host description file.	remote(5)
gethostent, sethostent, endhostent: get network	host entry. gethostbyname, gethostbyaddr,	gethostbyname(3N)
hosts:	host name data base	hosts(5)
phones: remote	host phone number data base	phones(5)
hostid: set or print identifier of current	host system.	hostid(1)
hostname: set or print name of current mkhosts: generate hashed	host system	hostname(1) mkhosts(8)
htable: convert NIC standard format	host tables.	htable(8)
gettable: get NIC format	host tables from a host.	gettable(8C)
system.	hostid: set or print identifier of current host	hostid(1)
L.aliases: UUCP	hostname alias file	L.aliases(5)
	hostname: set or print name of current host system	hostname(1)
nine and ICMP ECITO DECLIERE and the section is	hostnm: get name of current host	hostnm(3F)
ping: send ICMP ECHO_REQUEST packets to network	hosts: host name data base.	ping(8) hosts(5)
uptime: show	how long system has been up.	uptime(1)
format:	how to format disk packs.	format(8V)
lib2648: subroutines for the	HP 2648 graphics terminal	lib2648(3X)
	hp: MASSBUS disk interface	hp(4)
interface.	ht: TM-03/TE-16,TU-45,TU-77 MASSBUS magtape	ht(4)
host and network byte order.	htable: convert NIC standard format host tables htonl, htons, ntohl, ntohs: convert values between	htable(8)
and network byte order. htonl,	htons, ntohl, ntohs: convert values between host	byteorder(3N) byteorder(3N)
and network byte order. Intom,	hy: Network Systems Hyperchannel interface	hy(4)
asinh, acosh, atanh: inverse	hyperbolic functions	asinh(3M)
sinh, cosh, tanh:	hyperbolic functions	sinh(3M)
hy: Network Systems	Hyperchannel interface	hy(4)
value.	hypot, cabs: Euclidean distance, complex absolute	hypot(3M)
vacation: return	"I am on vacation" indication	vacation(1)
getarg, ibmemul:	iargc: return command line arguments	getarg(3F) ibmemul(4)
mset: retrieve ASCII to	IBM 3270 keyboard map.	mset(1)
mset: retrieve ASCII to	IBM 3270 keyboard map.	mset(1)
map3270: database for mapping ascii keystrokes into	IBM 3270 keys	map3270(5)
data base for mapping ASCII keystrokes into	IBM 3270 keys. map3270:	map3270(5)
cvt3812, cvt20to12, cvt00to12: convert IBM 3820 and	IBM 3800 fonts for use with the IBM 3812/	cvt3812(8)
IBM 3820 and IBM 3800 fonts for use with the	IBM 3812 Pageprinter. /cvt00to12: convert	cvt3812(8)
pprint: print text files on ppt: spooling system filter for the	IBM 3812 Pageprinter	pprint(1) ppt(8)
ptroff: print troff files on	IBM 3812 Pageprinter	ptroff(1)
vgrind: grind nice listings of programs for the	IBM 3812 Pageprinter	vgrind(1)
width3812: build width tables for	IBM 3812 Pageprinter fonts	width3812(8)
ibm3812pp:	1BM 3812 Pageprinter server	ibm3812pp(8)
printer3812:	IBM 3812 Pageprinter status information	printer3812(5)
3812/ cvt3812, cvt20to12, cvt00to12: convert colpro: column filter for	IBM 3820 and IBM 3800 fonts for use with the IBM IBM 4201 Proprinter	cvt3812(8) colpro(1)
ibmbit, ibmgra, ibmpro: output filters for the	IBM 4201 Proprinter and IBM 5152 Graphics Printer.	lpfilter(8r)
proff: nroff for the	IBM 4201 Proprinter and IBM 5152 Graphics Printer	proff(1)
post-processing filter. prfl:	IBM 4201 Proprinter/IBM 5152 Graphics Printer nroff	prfl(1)
ibm5081, mpel –	IBM 5081 Mega Pel Display interface	ibm5081(4)
ibm5151, mono:	IBM 5151 Monochrome Display interface	ibm5151(4)
output filters for the IBM 4201 Proprinter and proff: nroff for the IBM 4201 Proprinter and	IBM 5152 Graphics Printer. ibmbit, ibmgra, ibmpro: .	lpfilter(8r)
ibm5154, ega:	IBM 5152 Graphics Printer	proff(1) ibm5154(4)
interface. ibm6153, apa8:	IBM 6153 Advanced Monochrome Graphics Display	ibm6153(4)
ibm6154, apa8c:	IBM 6154 Advanced Color Graphics Display interface.	ibm6154(4)
interface. ibm6155, apa16:	IBM 6155 Extended Monochrome Graphics Display .	ibm6155(4)
ibm8513, ibm8514, ibm8604. ibm8514:	IBM 8514/A Display adapter for the ibm8503,	ibm8514(4)
scsiformat: format the	IBM 9332 disk unit.	scsiformat(8c)
Interface (SCSI) Adapter. sc:	IBM 9332 disks using the IBM Small Computer System	sc(4)
display. aedemul: graphics interfaces for the display interface. ibmaed, aed:	IBM Academic Information Systems experimental IBM Academic Information Systems experimental	aedemul(4) ibmaed(4)
display self-tests. aedtest:	IBM Academic Information Systems experimental	aedtest(8)
the image on a bitmap display and print it on an	IBM printer. bitprt: capture	bitprt(1)
debug: debugger for the	IBM RT PC	debug(8)
kbdlock: lock the keyboard of the	IBM RT PC	kbdlock(1)

un:	IBM RT PC Baseband Adapter for use with Ethernet	un(4)
lan:	IBM RT PC Token-Ring Adapter	lan(4)
sc: IBM 9332 disks using the landump: dump	IBM Small Computer System Interface (SCSI) Adapter. IBM Token-Ring Personal Computer Adapter	sc(4)
tn3270: full-screen remote login to	IBM VM/CMS	landump(8r) tn3270(1)
tn3270: full-screen remote login to	IBM VM/CMS	tn3270(1)
thozyo. Itali bolech remote login to	ibm3812pp: IBM 3812 Pageprinter server.	ibm3812pp(8)
interface.	ibm5081, mpel - IBM 5081 Mega Pel Display	ibm5081(4)
interface.	ibm5151, mono: IBM 5151 Monochrome Display	ibm5151(4)
interface.	ibm5154, ega: IBM 5154 Enhanced Graphics Display	ibm5154(4)
Graphics Display interface.	ibm6153, apa8: IBM 6153 Advanced Monochrome	ibm6153(4)
Display interface.	ibm6154, apa8c: IBM 6154 Advanced Color Graphics .	ibm6154(4)
Graphics Display interface.	ibm6155, apa16: IBM 6155 Extended Monochrome	ibm6155(4)
ibm8514: IBM 8514/A Display adapter for the	ibm8503, ibm8513, ibm8514, ibm8604	ibm8514(4)
tomosia. Ibiti ostajik bispiaj adapter for the	ibm8503, ibm8513, ibm8514, ibm8604	vga(4)
IBM 8514/A Display adapter for the ibm8503,	ibm8513, ibm8514, ibm8604. ibm8514:	ibm8514(4)
ibm8503,	ibm8513, ibm8514, ibm8604	vga(4)
ibm8503, ibm8513, ibm8514, ibm8604.	ibm8514: IBM 8514/A Display adapter for the	ibm8514(4)
8514/A Display adapter for the ibm8503, ibm8513,	ibm8514, ibm8604. ibm8514: IBM	ibm8514(4)
ibm8503, ibm8513,	ibm8514, ibm8604	vga(4)
Display adapter for the ibm8503, ibm8513, ibm8514,	ibm8604. ibm8514: IBM 8514/A	ibm8514(4)
ibm8503, ibm8513, ibm8514,	ibm8604	vga(4)
experimental display interface.	ibmaed, aed: IBM Academic Information Systems	ibmaed(4)
4201 Proprinter and IBM 5152 Graphics Printer.	ibmbit, ibmgra, ibmpro: output filters for the IBM	lpfilter(8r)
	ibmemul: IBM 3101 emulator.	ibmemul(4)
Proprinter and IBM 5152 Graphics Printer. ibmbit,	ibmgra, ibmpro: output filters for the IBM 4201	lpfilter(8r)
and IBM 5152 Graphics Printer. ibmbit, ibmgra,	ibmpro: output filters for the IBM 4201 Proprinter	lpfilter(8r)
	icheck: file system storage consistency check	icheck(8)
ping: send	ICMP ECHO REQUEST packets to network hosts	ping(8)
	icmp: Internet Control Message Protocol	icmp(4P)
setregid: set real and effective group	ID	setregid(2)
getpid: get process	id	getpid(3F)
setgid, setegid, setrgid: set user and group	ID. setuid, seteuid, setruid,	setuid(3)
whoami: print effective current user	id	whoami(1)
getuid, getgid: get user or group	ID of the caller	getuid(3F)
su: substitute user	id temporarily.	su(1)
form.	idate, itime: return date or time in numerical	idate(3F)
getpid, getppid: get process	identification.	getpid(2)
gethostid, sethostid: get/set unique	identifier of current host	gethostid(2)
hostid: set or print	identifier of current host system	hostid(1)
getgid, getegid: get group	identity.	getgid(2)
getuid, geteuid: get user	identity.	getuid(2)
setreuid: set real and effective user	idp: Xerox Internet Datagram Protocol	idp(4P) setreuid(2)
	ierrno: get system error messages.	perror(3F)
perror, gerror,	if: conditional statement.	csh(1)
biff: be notified	if mail arrives and who it is from.	biff(1)
eval, exec, exit, export, login,/ sh, for, case,	if, while, :, ., break, continue, cd,	sh(1)
hdh: ACC	IF-11/HDH IMP interface.	hdh(4)
non reco	isconfig: configure network interface parameters	ifconfig(8C)
	ifconfig: configure network interface parameters	ifconfig(8c)
unifdef: remove	ifdel'ed lines.	unifdef(1)
uu: TU58/DECtape	II UNIBUS cassette interface	uu(4)
	ik: Ikonas frame buffer, graphics device interface	ik(4)
ik:	Ikonas frame buffer, graphics device interface	ik(4)
	il: Interlan NI1010 10 Mb/s Ethernet interface	il(4)
scale: resize a bitmap	image	scale(1)
abort: terminate abruptly with memory	image	abort(3F)
VI_MImage, VI_FImage: draw an	image	image(3G)
xwd - X Window System, window	image dumper	xwd(1)
core: format of memory	image file.	core(5)
core: format of memory	image file.	core(5)
printer. bitprt: capture the	image on a bitmap display and print it on an IBM	bitprt(1)
xwud - X Window System, window	image undumper.	xwud(1)
gcore: get core	images of running processes	gcore(1)
notify: request	immediate notification	csh(1)
nohup: run command	immune to hangups	csh(1)
•	imp: 1822 network interface	imp(4)
	imp: IMP raw socket interface	imp(4P)
acc: ACC LH/DH	IMP interface.	acc(4)
css: DEC IMP-11A LH/DH	IMP interface.	css(4)
ddn: DDN Standard Mode X.25	IMP interface.	ddn(4)
hdh: ACC IF-11/HDH	IMP interface.	hdh(4)
implog:	IMP log interpreter	implog(8C)
implogd:	IMP logger process	implogd(8C)
imp:	IMP raw socket interface	imp(4P)

DEC	INAD 11 A LILIDIL INAD intenfere	/4\
css: DEC	IMP-11A LH/DH IMP interface	css(4)
xstr: extract strings from C programs to	implement shared strings implog: IMP log interpreter	xstr(1) implog(8C)
	implogd: IMP logger process.	implog(8C)
which: locate a program file	including aliases and paths (csh only).	which(1)
fsync: synchronize a file's	in-core state with that on disk.	fsync(2)
dump, dumpdates:	incremental dump format.	dump(5)
dump:	incremental file system dump.	dump(8)
restore:	incremental file system restore.	restore(8)
restore:	incremental file system restore.	restore(8)
indent:	indent and format C program source	indent(1)
	indent: indent and format C program source	indent(1)
tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal	independent operation routines. tgetent,	termcap(3X)
ptx: permuted	index	ptx(1)
L-dialcodes: UUCP phone number	index file	L-dialcodes(5)
bibliography. indxbib, lookbib: build inverted	index for a bibliography, find references in a	lookbib(1)
objects.	index, rindex, Inblnk, len: tell about character	index(3F)
strncat, strcmp, strncmp, strcpy, strncpy, strlen,	index, rindex: string operations. strcat,	string(3)
last:	indicate last logins of users and teletypes	last(1)
vacation: return "I am on vacation"	indication.	vacation(1)
syscall:	indirect system call	syscall(2)
fsplit: split a multi-routine Fortran file into	individual files	fsplit(1)
bibliography, find references in a bibliography.	indxbib, lookbib: build inverted index for a	lookbib(1)
	inet: Internet protocol family	inet(4F)
inet Inaof, inet netof: Internet address/	inet_addr, inet_network, inet_ntoa, inet_makeaddr,	inet(3N)
- · · -	inetd: internet "super - server"	inetd(8)
inet addr, inet network, inet ntoa, inet makeaddr,	inet Inaof, inet netof: Internet address/	inet(3N)
address/ inet addr, inet network, inet ntoa,	inet_makeaddr, inet_lnaof, inet_netof: Internet	inet(3N)
/inet network, inet ntoa, inet makeaddr, inet Inaof,	inet netof: Internet address manipulation routines	inet(3N)
inet netof: Internet address/ inet addr,	inet network, inet ntoa, inet makeaddr, inet lnaof,	inet(3N)
Internet address/ inet addr, inet network,	inet ntoa, inet makeaddr, inet lnaof, inet netof:	inet(3N)
/classdouble, classfloat, isnan, unordered, finite,	infinity, nextdouble, nextfloat, fptestround,/	ieee(3)
on a VAX (temporary).	infnan: signals invalid floating-point operations	infnan(3M)
bad144: read/write dec standard 144 bad sector	information	bad144(8)
dbx: dbx symbol table	information	dbx(5)
dbx: dbx symbol table	information	dbx(5)
dumpfs: dump file system	information	dumpfs(8)
pac: printer/plotter accounting	information	pac(8)
chfn, chsh, passwd: change password file	information	passwd(1)
printer3812: IBM 3812 Pageprinter status	information	printer3812(5)
support: hardware and software support	information	support(1)
(RVDs). /etc/rvd/rvdtab:	information about client Remote Virtual Disks	rvdtab(5)
getrusage: get	information about resource utilization	getrusage(2)
vtimes: get		vtimes(3C)
fstab: static	information about the filesystems	fstab(5)
man: find manual	information by keywords; print out the manual	man(1)
finger: user	information lookup program	finger(1)
XNSrouted: NS Routing	Information Protocol daemon.	XNSrouted(8C)
fingerd: remote user		fingerd(8C)
	information summarizer	xwininfo(1)
aedemul: graphics interfaces for the IBM Academic		aedemul(4)
ibmaed, aed: IBM Academic		ibmaed(4)
self-tests. aedtest: IBM Academic	Information Systems experimental display	aedtest(8)
	init: process control initialization.	init(8)
	init: process control initialization.	init(8)
inity manage control	initgroups: initialize group access list	initgroups(3)
init: process control	initialization.	init(8)
init: process control	initialization.	init(8)
tset: terminal dependent ioinit: change f77 I/O	initialization.	tset(1)
		ioinit(3F)
ttys: terminal	initialization data	ttys(5)
VI_Init, VI_Term:	initialize group access list	init(3G)
initgroups: xinit - X window system	initializer	initgroups(3)
connect:	initiate a connection on a socket.	xinit(1) connect(2)
popen, pclose:	initiate I/O to/from a process	popen(3)
generator; routines for changing/ random, srandom,	initiate, setstate: better random number	random(3)
flmin, flmax, ffrac, dflmin, dflmax, dffrac,	inmax: return extreme values	flmin(3F)
clri: clear	i-node	clri(8)
fs,	inode: format of file system volume.	fs(5)
		* 1
read, readv: read soelim: eliminate .so's from nroff	input	read(2) soelim(1)
scanf, fscanf, sscanf: formatted	input conversion.	scanf(3S)
events. xemul: X	input emulator for queuing keyboard and mouse	xemul(4)
ungetc: push character back into		~~!!!#[[~]
	•	
fread, fwrite: buffered binary	input stream	ungetc(3S) fread(3S)

stdio: standard buffered	input/output package	stdio(3S)
ferror, feof, clearerr, fileno: stream status	inquiries	ferror(3S)
refer: find and	insert literature references in documents insert/remove element from a queue	refer(1) insque(3)
insque, remque:	insque, remque: insert/remove element from a queue	insque(3)
ptfinstall:	install a Program Temporary Fix (PTF)	ptfinstall(8)
install:	install binaries.	install(1)
restore.tape, restore.net:	install: install binaries:	install(1) restore.tape(8)
learn: computer aided	instruction about UNIX	learn(1)
learn: computer aided	instruction about UNIX	learn(1)
fsck: file system consistency check and	interactive repair.	fsck(8)
ace: ACC LH/DH IMP asy: multi-port asynchronous communications RS232C	interface	acc(4) asy(4)
cons: VAX-11 console	interface	cons(4)
cons: keyboard and console display	interface	cons(4)
crl: VAX 8600 console RL02	interface	crl(4)
css: DEC IMP-11A LH/DH IMP ct: phototypesetter	interface	css(4) ct(4)
ddn: DDN Standard Mode X.25 IMP	interface.	ddn(4)
de: DEC DEUNA 10 Mb/s Ethernet	interface	de(4)
dn: DN-11 autocall unit	interface	dn(4)
ec: 3Com 10 Mb/s Ethernet	interface	ec(4)
en: Xerox 3 Mb/s Ethernet ex: Excelan 10 Mb/s Ethernet	interface	en(4) ex(4)
fd: diskette	interface.	fd(4)
fl: console floppy	interface	fl(4)
hd: hard disk	interface	hd(4)
hdh: ACC IF-11/HDH IMP	interface	hdh(4)
hp: MASSBUS disk ht: TM-03/TE-16,TU-45,TU-77 MASSBUS magtape	interface	hp(4) ht(4)
hy: Network Systems Hyperchannel	interface.	hy(4)
ibm5081, mpel - IBM 5081 Mega Pel Display	interface	ibm5081(4)
ibm5151, mono: IBM 5151 Monochrome Display	interface	ibm5151(4)
ibm5154, ega: IBM 5154 Enhanced Graphics Display apa8: IBM 6153 Advanced Monochrome Graphics Display	interface	ibm5154(4) ibm6153(4)
apa8c: IBM 6154 Advanced Color Graphics Display	interface. ibm6154,	ibm6154(4)
IBM 6155 Extended Monochrome Graphics Display	interface. ibm6155, apa16:	ibm6155(4)
Academic Information Systems experimental display	interface. ibmaed, aed: IBM	ibmaed(4)
ik: Ikonas frame buffer, graphics device il: Interlan N11010 10 Mb/s Ethernet	interface	ik(4) il(4)
imp: 1822 network	interface	imp(4)
imp: IMP raw socket	interface	imp(4P)
ix: Interlan Np100 10 Mb/s Ethernet	interface	ix(4)
lo: software loopback network mouse: mouse	interface	lo(4) mouse(4)
mt: TM78/TU-78 MASSBUS magtape	interface	mt(4)
mtio: UNIX magtape	interface	mtio(4)
mtio: 4.3/RT magtape	interface	mtio(4)
np: Interlan Np100 10 Mb/s Ethernet pcl: DEC CSS PCL-11 B Network	interface	np(4)
pci: DEC CSS PCE-II B Network plot: graphics	Interface.	pcl(4) plot(5)
and Sutherland Picture System 2 graphics device	interface. ps: Evans	ps(4)
psp: planar serial port RS232C	interface	psp(4)
	interface	qe(4)
rx: DEC RX02 floppy disk speaker: console speaker	interface	rx(4) speaker(4)
st: streaming-tape	interface.	st(4)
tm: TM-11/TE-10 magtape	interface	tm(4)
tmscp: DEC TMSCP magtape	interface	tmscp(4)
ts: TS-11 magtape tty: general terminal	interface	ts(4) tty(4)
tty: general terminal		tty(4)
tu: VAX-11/730 and VAX-11/750 TU58 console cassette	interface	tu(4)
uda: UDA-50 disk controller	interface	uda(4)
VI_Term: initialize and terminate the subroutine cont, point, linemod, space, closepl: graphics	interface. VI_Init,	init(3G) plot(3X)
ut: UNIBUS TU45 tri-density tape drive	interface	ut(4)
uu: TU58/DECtape II UNIBUS cassette	interface	uu(4)
va: Benson-Varian	interface	va(4)
vp: Versatec nsip: software network	interface	vp(4) nsip(4)
Xlib: C Language X Window System	Interface Library.	Xlib(3X)
ifconfig: configure network	interface parameters	ifconfig(8C)
ifconfig: configure network	interface parameters	ifconfig(8c)
syscall: system call	interface program	syscall(8)

Internal to the state of the st		
IBM 9332 disks using the IBM Small Computer System	Interface (SCSI) Adapter. sc:	sc(4)
plot: openpl et al.: f77 library	interface to plot (3X) libraries	plot(3F)
fpa: direct	0.1	fpa(3X)
xmh: X window	interface to the mh Mail Handler	xmh(1)
nfcomment: a user	interface to the notesfile system	nfcomment(3)
slattach: attach serial lines as network	interfaces.	
experimental display. aedemul: graphics	interfaces for the IBM Academic Information Systems	aedemul(4)
il:	Interlan NI1010 10 Mb/s Ethernet interface	
ix:	Interlan Np100 10 Mb/s Ethernet interface	ix(4)
np:	Interlan Np100 10 Mb/s Ethernet interface	np(4)
swapon: add a swap device for	interleaved paging/swapping	swapon(2)
sendmail: send mail over the	internet	sendmail(8)
/inet_ntoa, inet_makeaddr, inet_lnaof, inet_netof:	Internet address manipulation routines	inet(3N)
icmp:	Internet Control Message Protocol	icmp(4P)
	Internet Datagram Protocol	idp(4P)
	Internet domain name server	named(8)
	Internet File Transfer Protocol server	ftpd(8C)
· · · · · · · · · · · · · · · · · · ·	Internet Protocol.	ip(4P)
inet:	Internet protocol family.	inet(4F)
inetd:	internet "super – server".	inetd(8)
tcp:	Internet Transmission Control Protocol	tcp(4P)
udp:	Internet User Datagram Protocol	udp(4P)
•		
whois: DARPA	Internet user name directory service	whois(1)
spline:	interpolate smooth curve.	spline(1G)
implog: IMP log	interpreter	implog(8C)
lisp: lisp	interpreter	lisp(1)
px: Pascal	interpreter	px(1)
uuxqt: UUCP execution file	interpreter	uuxqt(8C)
pix: Pascal	interpreter and executor	pix(1)
pi: Pascal	interpreter code translator	pi(1)
csh: a shell (command	interpreter) with C-like syntax	csh(1)
pipe: create an	interprocess communication channel	pipe(2)
atomically release blocked signals and wait for	interrupt. sigpause:	sigpause(2)
siginterrupt: allow signals to	interrupt system calls	siginterrupt(3)
onintr: process	interrupts in command scripts	csh(1)
intro:	introduction to C library functions	intro(3)
intro:	introduction to commands	intro(1)
intro:	introduction to display graphics subroutines.	intro(3G)
intro:	introduction to FORTRAN library functions	intro(3F)
math:	introduction to mathematical library functions	math(3M)
networking:	introduction to networking facilities	intro(4N)
intro:	introduction to special files and hardware support	
intro:		intro(4)
	introduction to special files and hardware support	intro(4)
intro:	introduction to system calls and error numbers	intro(2)
intro:	introduction to system calls and error numbers	intro(2)
commands. intro:	introduction to system maintenance and operation	intro(8)
commands. intro:	introduction to system maintenance and operation	intro(8)
ncheck: generate names from	i-numbers.	ncheck(8)
(temporary). infnan: signals	invalid floating-point operations on a VAX	infnan(3M)
	inverse hyperbolic functions	asinh(3M)
atan, atan2: trigonometric functions and their	inverses. sin, cos, tan, asin, acos,	sin(3M)
m_in, mout,/ madd, msub, mult, mdiv, pow, gcd,	invert, rpow, msqrt, mcmp, move, min, omin, fmin,	mp(3X)
in a bibliography. indxbib, lookbib: build	inverted index for a bibliography, find references	lookbib(1)
tread, twrite, trewin, tskipf, tstate: f77 tape	I/O. topen, tclose,	topen(3F)
bus: control of access to the system	I/O bus	bus(4)
ioinit: change f77	I/O initialization.	ioinit(3F)
select: synchronous	I/O multiplexing	select(2)
iostat: report	I/O statistics	iostat(1)
popen, pclose: initiate	I/O to/from a process	popen(3)
	ioctl: control device	ioctl(2)
	ioinit: change f77 I/O initialization	ioinit(3F)
	iostat: report I/O statistics	iostat(1)
	ip: Internet Protocol	ip(4P)
network interface encapsulating ns packets in	ip packets nsip: software	nsip(4)
rand, drand,	irand: return random values	rand(3F)
random, drandm,	irandm: better random number generator	random(3F)
isalpha, isupper, islower, isdigit, isxdigit,	isalnum, isspace, ispunct, isprint, isgraph,/	ctype(3)
isalnum, isspace, ispunct, isprint, isgraph,/	isalpha, isupper, islower, isdigit, isxdigit,	ctype(3)
/isspace, ispunct, isprint, isgraph, iscntrl,	isascii, toupper, tolower, toascii: character/	* * 1
		ctype(3)
ttynam,	isatty: find name of a terminal port	ttynam(3F)
ttyname,	isatty, ttyslot: find name of a terminal	ttyname(3)
/isalnum, isspace, ispunct, isprint, isgraph,	iscntrl, isascii, toupper, tolower, toascii:/	ctype(3)
isprint, isgraph,/ isalpha, isupper, islower,	isdigit, isxdigit, isalnum, isspace, ispunct,	ctype(3)
/isxdigit, isalnum, isspace, ispunct, isprint,	isgraph, iscntrl, isascii, toupper, tolower,/	ctype(3)
ispunct, isprint, isgraph,/ isalpha, isupper,	islower, isdigit, isxdigit, isalnum, isspace,	ctype(3)
drem, logb, scalb, rint, classdouble, classfloat,	isnan, unordered, finite, infinity, nextdouble,/	ieee(3)

/isdigit, isxdigit, isalnum, isspace, ispunct,	isprint, isgraph, iscntrl, isascii, toupper,/	ctype(3)
/islower, isdigit, isxdigit, isalnum, isspace,	ispunct, isprint, isgraph, iscntrl, isascii,/	ctype(3)
/isupper, islower, isdigit, isxdigit, isalnum,	isspace, ispunct, isprint, isgraph, iscntrl,/	* * * *
system:	issue a shell command	
isspace, ispunct, isprint, isgraph,/ isalpha,	isupper, islower, isdigit, isxdigit, isalnum,	ctype(3)
isgraph,/ isalpha, isupper, islower, isdigit,	isxdigit, isalnum, isspace, ispunct, isprint,	ctype(3)
idate,	itime: return date or time in numerical form	idate(3F)
omin, fmin, m_in, mout, omout, fmout, m_out, sdiv,	itom: multiple precision integer arithmetic. /min,	1 1 7
	ix: Interlan Np100 10 Mb/s Ethernet interface	
	j0, j1, jn, y0, y1, yn: Bessel functions	
	j1, jn, y0, y1, yn: Bessel functions	
j0, j1,	jn, y0, y1, yn: Bessel functions.	
bg: place	job in background	csh(1)
fg: bring jobs: print current		
stop: halt a	job or process.	
kill: kill		
lprm: remove	jobs from the line printer spooling queue	lprm(1)
ipini. ieniove	jobs: print current job list.	
atrm: remove	jobs spooled by at	
	jobs waiting to be run.	
ard, print the queue of	join: relational database operator.	
msgs: system messages and	junk mail program.	
mogo. of stern messages and	kbdemul: default keyboard emulator.	
	kbdlock: lock the keyboard of the IBM RT PC	
bufemul:	kernel buffering emulator.	
makekey: generate encryption	key.	
cons:	keyboard and console display interface	
xemul: X input emulator for queuing	keyboard and mouse events	
kbdemul: default	keyboard emulator	
mset: retrieve ASCII to IBM 3270	keyboard map.	mset(1)
mset: retrieve ASCII to IBM 3270	keyboard map	mset(1)
xmodmap, xprkbd - X Window System	keyboard modifier utilities	xmodmap(1)
kbdlock: lock the	keyboard of the IBM RT PC	kbdlock(1)
pf: set	keyboard program-function keys	
keyboard_codes:	keyboard scancode table	
	keyboard_codes: keyboard scancode table	keyboard_codes(5)
database for mapping ascii keystrokes into IBM 3270	keys. map3270:	
base for mapping ASCII keystrokes into IBM 3270	keys. map3270: data	
pf: set keyboard program-function	keys	pf(1)
map3270: database for mapping ascii	keystrokes into IBM 3270 keys	
map3270: data base for mapping ASCII	keystrokes into IBM 3270 keys	map3270(5)
apropos: locate commands by	keyword lookup.	
man: find manual information by	keywords; print out the manual	
	kg: KL-11/DL-11W line clock.	kg(4)
profile buffers.	kgmon: generate a dump of the operating system's	
Kill:	kill jobs and processes.	csh(1)
	kill: kill jobs and processes.	
	kill: send a signal to a process.	kill(3F)
	kill: send signal to a process.	
	kill: terminate a process with extreme prejudice	
hassal functions, of two	killpg: send signal to a process group	
bessel functions: of two	KL-11/DL-11W line clock.	, ,
kg: memory. mem,		O()
mem.		
mem, kmem,		mem(4)
mem, kmem, kmem1,		
mem, kmem, kmem1, kmem2,		
linemod, space, closepl:/ plot: openpl, erase,		
	L. aliases: UUCP hostname alias file	
	lan: IBM RT PC Token-Ring Adapter	lan(4)
Adapter.	landump: dump IBM Token-Ring Personal Computer .	
awk: pattern scanning and processing		
bc: arbitrary-precision arithmetic	language.	_ ` '
esi: Extended Fortran	Language.	
set, shift, times, trap, umask, wait: command	language. /exit, export, login, read, readonly,	
fp: Functional Programming	language compiler/interpreter	
vgrindefs: vgrind's	language definition data base	
Xlib: C	, ,	, , ,
order.	lastcomm: show last commands executed in reverse	
	L.cmds: UUCP remote command permissions file	
	ld: link editor.	\ <i>\</i>
		` '
	L-devices: UUCP device description file ldexp, modf: split into mantissa and exponent	
nexp,	mone, moun spin mo manussa and caponent	··cxp(o)

	L-dialcodes: UUCP phone number index file L-dialcodes(5)
	learn: computer aided instruction about UNIX learn(1) learn: computer aided instruction about UNIX learn(1)	
leave: remind you when you have to	leave leave(1)	
	leave: remind you when you have to leave leave(1)	
exit: getusershell, setusershell, endusershell: get	leave shell	/2 \
index, rindex, lnblnk,	len: tell about character objects index(3F)	3)
truncate, ftruncate: truncate a file to a specified	length truncate(2)	
getopt: get option	letter from argv getopt(3)	
rvdchlog: change logging	level of Remote Virtual Disk (RVD) server rvdchlog(8)	
lex: generator of	lex: generator of lexical analysis programs lex(1) lexical analysis programs lex(1)	
8	lgamma: log gamma function lgamma(3M)
acc: ACC	LH/DH IMP interface acc(4)	
css: DEC IMP-11A terminal.	LII/DII IMP interface	
ranlib: convert archives to random	lib2648: subroutines for the HP 2648 graphics lib2648(3X) libraries ranlib(1)	
et al.: f77 library interface to plot (3X)	libraries plot: openpl plot(3F)	
lorder: find ordering relation for an object	library lorder(1)	
Xlib: C Language X Window System Interface	Library Xlib(3X)	
ar: archive intro: introduction to C	(library) file format	
intro: introduction to FORTRAN	library functions intro(3F)	
math: introduction to mathematical	library functions math(3M)	
plot: openpl et al.: f77	library interface to plot (3X) libraries plot(3F)	
ar: archive and	library maintainer	
limit: alter per-process resource	limit: alter per-process resource limitations	
unlimit: remove resource	limitiations	
quota: display disc usage and	limits quota(1)	
VI_ALine, VI_RLine: draw a	line line(3G)	
getarg, iargc: return command space, closepl:/ plot: openpl, erase, label,	line arguments getarg(3F) line, circle, arc, move, cont, point, linemod, plot(3X)	
kg: KL-11/DL-11W	line clock	
VI_Dash: set	line dash pattern dash(3G)	
ap: asynchronous data mode protocol	line discipline ap(4)	
tb: tb:	line discipline for digitizing devices	
(obsolete). bk:	line discipline for digitizing devices	
col: filter reverse	line feeds col(1)	
sysline: display system status on status	line of a terminal sysline(1)	
lpr: off	line print	
lp: lp:	line printer	
lpc:	line printer control program	
lpd:	line printer daemon	
lprm: remove jobs from the	line printer spooling queue lprm(1)	
VI_Width: set /erase, label, line, circle, arc, move, cont, point,	line width	
lptest: generate	lineprinter ripple pattern lptest(1)	
head: give first few	lines head(1)	
unifdef: remove ifdef'ed	lines unifdef(1)	
slattach: attach serial comm: select or reject	lines as network interfaces slattach(8C) lines common to two sorted files comm(1)	
fold: fold long	lines for finite width output device fold(1)	
uniq: report repeated	lines in a file uniq(1)	
look: find	lines in a sorted list look(1)	
rev: reverse	lines of a file rev(1)	
readlink: read value of a symbolic ld:	link	
ld:	link editor	
a.out: assembler and	link editor output a.out(5)	
a.out: assembler and	link editor output a.out(5)	
	link: make a hard link to a file link(2) link: make a link to an existing file link(3F)	
link: make a hard	link to a file link(2)	
symlink: make symbolic	link to a file symlink(2)	
link: make a	link to an existing file link(3F)	
ln: make	links	
lyref	lint: a C program verifier lint(1)	
lxref: lisp:	lint: a C program verifier lint(1) lisp cross reference program lxref(1)	
	lint: a C program verifier lint(1) lisp cross reference program lxref(1)	
	lint: a C program verifier	

•		
glob: filename expand argument	list	csh(1)
history: print history event	list	csh(1)
jobs: print current job	list	csh(1)
shift: manipulate argument	list	csh(1)
getgroups: get group access	list	getgroups(2)
look: find lines in a sorted	list.	look(1)
nm: print name	list.	nm(1)
setgroups: set group access	list.	setgroups(2)
	list.	symorder(1)
symorder: rearrange name		
initgroups: initialize group access	list.	initgroups(3)
nlist: get entries from name	list	nlist(3)
varargs: variable argument	list.	varargs(3)
vdstats:	list client Remote Virtual Disk (RVD) statistics	vdstats(8)
ls:	list contents of directory	ls(1)
xlsfonts - X window system font	list displayer	xisfonts(1)
foreach: loop over	list of names	csh(1)
rvdhosts: print a	list of RVD servers.	rvdhosts(8)
users: compact	list of users who are on the system.	users(1)
listen:	listen for connections on a socket.	1.1
. nsten.		listen(2)
	listen: listen for connections on a socket	listen(2)
vgrind: grind nice	listings of programs.	vgrind(1)
vgrind: grind nice	listings of programs for the IBM 3812 Pageprinter	vgrind(1)
	liszt: compile a Franz Lisp program	liszt(1)
refer: find and insert	literature references in documents	refer(1)
	In: make links.	ln(1)
index, rindex,	Inblnk, len: tell about character objects	index(3F)
maex, maex,		
	lo: software loopback network interface	lo(4)
xload - X window system	load average display.	xload(1)
Point Accelerator, afpacode:	load, test, and bring online the Advanced Floating	afpacode(8r)
	loc: return the address of an object	loc(3F)
and time to ASCII. ctime,	localtime, gmtime, asctime, timezone: convert date	ctime(3)
(csh only). which:	locate a program file including aliases and paths	which(1)
apropos:	locate commands by keyword lookup.	apropos(1)
• •		
whereis:	locate source, binary, and or manual for program	whereis(1)
end, etext, edata: last	locations in program.	end(3)
flock: apply or remove an advisory	lock on an open file	flock(2)
	lock: reserve a terminal	lock(1)
kbdlock:	lock the keyboard of the IBM RT PC	kbdlock(1)
collect system diagnostic messages to form error	log. dmesg:	dmesg(8)
logger: make entries in the system	log	logger(1)
openlog, closelog, setlogmask: control system	log. syslog,	syslog(3)
aedjournal: display commands in a	log file.	aedjournal(1)
aedrunner: execute graphics commands in a	log file.	aedrunner(1)
begin logging subroutine calls and close a	log file. VI_Login, VI_Logout:	log(3G)
VI_Run: process a	log file.	run(3G)
lgamma:	log gamma function	lgamma(3M)
implog: IMP	log interpreter.	implog(8C)
nfabort: dump core and	log it in a notesfile.	nfabort(3)
power. exp, expm1,	log, log10, log1p, pow: exponential, logarithm,	exp(3M)
rvdlog: cause Remote Virtual Disk (RVD) server to		rvdlog(8)
	log statistics.	
syslogd:	log systems messages.	syslogd(8)
syslogd:	log systems messages.	syslogd(8)
	log: tell Venus about your password	log(1)
exp, expm1, log,	log10, log1p, pow: exponential, logarithm, power	exp(3M)
exp, expm1, log, log10,	log1p, pow: exponential, logarithm, power	exp(3M)
exp, expm1, log, log10, log1p, pow: exponential,	logarithm, power	exp(3M)
manipulations. copysign, drem, finite,	logb, scalb: copysign, remainder, exponent	ieee(3M)
unordered, finite, infinity,/ copysign, drem,	logb, scalb, rint, classdouble, classfloat, isnan,	ieee(3)
unordered, mine, mining, copysign, drein,	logger: make entries in the system log	logger(1)
!1.a.d. 13.6D	, .	
implogd: IMP	logger process.	implogd(8C)
rvdchlog: change	logging level of Remote Virtual Disk (RVD) server	rvdchlog(8)
VI_Login, VI_Logout: begin	logging subroutine calls and close a log file	log(3G)
flush: flush output to a	logical unit.	flush(3F)
fseek, ftell: reposition a file on a	logical unit.	fseek(3F)
getc, fgetc: get a character from a	logical unit.	getc(3F)
putc, fputc: write a character to a fortran	logical unit.	putc(3F)
ac:	login accounting.	ac(8)
	login: login new user.	csh(1)
getlog: get user's	login name	getlog(3F)
getlogin: get	login name	getlogin(3)
login:	login new user.	csh(1)
/break, continue, cd, eval, exec, exit, export,	login, read, readonly, set, shift, times, trap,/	sh(1)
utmp, wtmp:	login records.	utmp(5)
	login server.	rlogind(8C)
rlogind: remote	HINGH SPI VPI	
	login: sign on	login(1)
tn3270: full-screen remote		

tn3270: full-screen remote	login to IBM VM/CMS	tn3270(1)
last: indicate last	logins of users and teletypes.	last(1)
indiana indiana	logout: end session.	
setjmp,	longimp: non-local goto.	
ovijinp;	look: find lines in a sorted list	
find references in a bibliography, indxbib,	lookbib: build inverted index for a bibliography,	look(1)
apropos: locate commands by keyword	lookup	apropos(1)
finger: user information	lookup program.	
	•	finger(1)
break: exit while/foreach	loop	csh(1)
continue: cycle in	loop	csh(1)
end: terminate	loop.	
foreach:	loop over list of names	csh(1)
lo: software	loopback network interface	lo(4)
library.	lorder: find ordering relation for an object	lorder(1)
mklost + found: make a	lost + found directory for fsck	mklost + found(8)
	lp: line printer	lp(4)
	lp: line printer	lp(4)
	lpc: line printer control program.	lpc(8)
	lpd: line printer daemon	lpd(8)
	lpq: spool queue examination program	lpq(1)
	lpr: off line print	lpr(1)
queue.	lprm: remove jobs from the line printer spooling	lprm(1)
1	lptest: generate lineprinter ripple pattern	lptest(1)
	ls: list contents of directory.	ls(1)
	lseek: move read/write pointer.	lseek(2)
bit: and, or, xor, not, rshift,	Ishift bitwise functions.	bit(3F)
stat.	Istat, fstat: get file status.	
stat,	Istat, fstat: get file status.	• • • • • • • • • • • • • • • • • • •
Stat,	L.sys: UUCP remote host description file.	
time atime	ltime, gmtime: return system time.	time(3F)
time, ctime,	lxref: lisp cross reference program	
		lxref(1)
404-4	m4: macro processor	
getfloatstate: return		
bk: line discipline for	machine-machine communication (obsolete)	bk(4)
m4:	macro processor.	m4(1)
alias: shell	macros	csh(1)
toupper, tolower, toascii: character classification	macros. /isprint, isgraph, iscntrl, isascii,	ctype(3)
msqrt, mcmp, move, min, omin, fmin, m_in, mout,/	madd, msub, mult, mdiv, pow, gcd, invert, rpow,	mp(3X)
tcopy: copy a	mag tape	tcopy(1)
mt:	magnetic tape manipulating program	mt(1)
mt:	magnetic tape manipulating program	mt(1)
ht: TM-03/TE-16,TU-45,TU-77 MASSBUS	magtape interface	ht(4)
mt: TM78/TU-78 MASSBUS	magtape interface	mt(4)
mtio: UNIX	magtape interface	mtio(4)
mtio: 4.3/RT	magtape interface	mtio(4)
tm: TM-11/TE-10	magtape interface	tm(4)
tmscp: DBC TMSCP	magtape interface.	tmscp(4)
ts: TS-11	magtape interface.	
rmt: remote	magtape protocol module.	` '
mail: send and receive	mail	mail(1)
xsend, xget, enroll: secret	mail.	xsend(1)
sendbug:	mail a system bug report to 4bsd-bugs.	sendbug(1)
newaliases: rebuild the data base for the	mail aliases file.	newaliases(1)
binmail: send or receive		
biff: be notified if	mail among users	binmail(1) biff(1)
from: who is my	mail from?	from(1)
		` '
xmh: X window interface to the mh	Mail Handler	xmh(1)
sendmail: send	mail over the internet	sendmail(8)
msgs: system messages and junk		msgs(1)
rmail: handle remote	mail received via uucp.	rmail(1)
· · · · · · · · · · · · · · · · · · ·	mail: send and receive mail	mail(1)
mem, kmem:		mem(4)
mem, kmem, kmem1, kmem2, kmem4, ros, afpamem:	main memory.	mem(4)
make:	maintain program groups	make(1)
ar: archive and library	maintainer	ar(1)
intro: introduction to system	maintenance and operation commands	intro(8)
intro: introduction to system	maintenance and operation commands	intro(8)
fdisk: boot record partition table	maintenance utility	fdisk(8)
minidisk: minidisk	maintenance utility.	minidisk(8r)
mkdir:	make a directory.	mkdir(1)
mkdir:	make a directory file	mkdir(2)
link:	make a hard link to a file	link(2)
link:		link(3F)
mklost + found:	make a lost + found directory for fsck	
mknod:	make a special file	mknod(2)
mktemp:	make a unique file name.	

•		• (0)
makesym:	make debugger symbol table	makesym(8)
logger:	make entries in the system log	logger(1)
in.	make links	in(1)
arimlinle.	make: maintain program groups	make(1)
symlink: makedev:		symlink(2)
	make system special files	makedev(8)
makedev:	make system special files	makedev(8)
vwidth:		vwidth(1)
script:	make typescript of terminal session.	script(1)
	makedev: make system special files	makedev(8)
	makedev: make system special files	makedev(8)
	makekey: generate encryption key	makekey(8)
	makesym: make debugger symbol table	makesym(8)
	malloc, free, falloc: memory allocator.	malloc(3F)
allocator.	malloc, free, realloc, calloc, alloca: memory	malloc(3)
the manual.	man: find manual information by keywords; print out .	man(1)
vddb: Remote Virtual Disk (RVD) data base	manager	vddb(8)
wm: a simple real-estate-driven window	manager.	wm(1)
.PP uwm - Window	Manager Client Application of X.PP	uwm(1)
shift:	manipulate argument list	csh(1)
quota:	manipulate disk quotas	quota(2)
VI_Font, VI_GetFont, VI_DropFont: select and	manipulate fonts.	font(3G)
tp:	manipulate tape archive.	tp(1)
route: manually	manipulate the routing tables	route(8C)
mt: magnetic tape	manipulating program	mt(1)
mt: magnetic tape	manipulating program	mt(1)
inet_Inaof, inet_netof: Internet address	manipulation routines. /inet_ntoa, inet_makeaddr,	inet(3N)
finite, logb, scalb: copysign, remainder, exponent	manipulations. copysign, drem,	ieee(3M)
frexp, ldexp, modf: split into	mantissa and exponent	frexp(3)
catman: create the cat files for the	manual	catman(8)
find manual information by keywords; print out the	manual, man:	man(1)
whereis: locate source, binary, and or	manual for program	whereis(1)
manual, man: find	manual information by keywords; print out the	man(1)
route:	manually manipulate the routing tables	route(8C)
into IBM 3270 keys.	map3270: data base for mapping ASCII keystrokes	map3270(5)
IBM 3270 keys.	map3270: database for mapping ascii keystrokes into	map3270(5)
map3270: database for	mapping ascii keystrokes into IBM 3270 keys	map3270(5)
map3270: data base for	mapping ASCII keystrokes into IBM 3270 keys	map3270(5)
umask: change or display file creation	mask.	csh(1)
sigsetmask: set current signal	mask	sigsetmask(2)
umask: set file creation mode	mask	umask(2)
mkstr: create an error message file by	massaging C source	mkstr(1)
hp:	MASSBUS disk interface.	hp(4)
ht: TM-03/TE-16,TU-45,TU-77	MASSBUS magtape interface	ht(4)
mt: TM78/TU-78	MASSBUS magtape interface	mt(4)
functions.	math: introduction to mathematical library	math(3M)
math: introduction to	mathematical library functions.	math(3M)
eqn, neqn, checkeq: typeset	mathematics	eqn(1)
getrlimit, setrlimit: control	maximum system resource consumption	getrlimit(2)
vlimit: control	maximum system resource consumption	vlimit(3C)
de: DEC DEUNA 10		de(4)
	Mb/s Ethernet interface	` '
ec: 3Com 10	Mb/s Ethernet interface	ec(4)
en: Xerox 3		
	Mb/s Ethernet interface	en(4)
ex: Excelan 10	Mb/s Ethernet interface	en(4) ex(4)
il: Interlan NI1010 10	Mb/s Ethernet interface. Mb/s Ethernet interface.	en(4) ex(4) il(4)
il: Interlan NI1010 10 ix: Interlan Np100 10	Mb/s Ethernet interface. Mb/s Ethernet interface. Mb/s Ethernet interface.	en(4) ex(4) il(4) ix(4)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10	Mb/s Ethernet interface.	en(4) ex(4) il(4) ix(4) np(4)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10	Mb/s Ethernet interface.	en(4) ex(4) il(4) ix(4) np(4) qe(4)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt,	Mb/s Ethernet interface. mcmp, move, min, omin, fmin, m_in, mout, omout,/	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult,	Mb/s Ethernet interface. mcmp, move, min, omin, fmin, m_in, mout, omout,/ mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move, .	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081	Mb/s Ethernet interface. mcmp, move, min, omin, fmin, m_in, mout, omout,/ mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move, Mega Pel Display interface.	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10	Mb/s Ethernet interface. Mcmp, move, min, omin, fmin, m_in, mout, omout,/ mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move, Mega Pel Display interface. Megabit ring.	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081	Mb/s Ethernet interface. Mcmp, move, min, omin, fmin, min, mout, omout,/ mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move, Mcga Pel Display interface. Megabit ring. mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) mem(4)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, god, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory.	Mb/s Ethernet interface. Mcmp, move, min, omin, fmin, m_in, mout, omout,/ mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move, Mega Pel Display interface. Megabit ring. Megabit ring. mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main mem, kmem: main memory.	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) mem(4)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group	Mb/s Ethernet interface. Mcgnp, move, min, omin, fmin, m_in, mout, omout,/ mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move, Mcga Pel Display interface. Megabit ring. Mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main mem, kmem: main memory. memberships.	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) mem(4) groups(1)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group mem, kmem: main	Mb/s Ethernet interface. Mcga Pel Display interface. Mcga Pel Display interface. Mcga Pel Display interface. Mcgabit ring. mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main mem, kmem: main memory. memberships. memory.	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) mem(4) groups(1) mem(4)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group mem, kmem: main mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main	Mb/s Ethernet interface. Mcga Pel Display interface. Mcga Pel Display interface. Megabit ring. mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main mem, kmem: main memory. memberships. memory.	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) nem(4) groups(1) mem(4) mem(4)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout./ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group mem, kmem: main mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main malloc, free, realloc, calloca:	Mb/s Ethernet interface. Mcmp, move, min, omin, fmin, m_in, mout, omout,/ mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move, Mcga Pel Display interface. Megabit ring. mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main mem, kmem: main memory. memberships. memory. memory. memory.	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) mem(4) groups(1) mem(4) mem(4) mem(4) malloc(3)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group mem, kmem: main mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main malloc, free, realloc, calloc, alloca: malloc, free, falloc:	Mb/s Ethernet interface. Mcmp, move, min, omin, fmin, m_in, mout, omout,/ mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move, Mcga Pel Display interface. Megabit ring. mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main mem, kmem: main memory. memberships. memory. memory. memory allocator. memory allocator.	en(4) ex(4) ii(4) ix(4) np(4) qe(4) mp(3X) ibm5081(4) vv(4) n mem(4) groups(1) mem(4) mem(4) mem(4) malloc(3) malloc(3F)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout./ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main malloc, free, realloc, calloc, alloca: malloc, free, falloc: valloc: aligned	Mb/s Ethernet interface. Mcmp, move, min, omin, fmin, m_in, mout, omout,/ mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move, Mcga Pel Display interface. Megabit ring. mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main mem, kmem: main memory. memberships. memory. memory. memory allocator. memory allocator.	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) ibm5081(4) vv(4) i mem(4) groups(1) mem(4) mem(4) malloc(3) malloc(3F) valloc(3C)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group mem, kmem: main mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main malloc, free, realloc, calloca, alloca: malloc, free, falloc: valloc: aligned dumpaed: dump aed display	Mb/s Ethernet interface. McMs Ethernet inte	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) mem(4) groups(1) mem(4) mem(4) mem(4) malloc(3) malloc(3F) valloc(3C) dumpaed(1)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group mem, kmem: main mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main malloc, free, realloc, calloc, alloca: malloc, free, falloc: valloc: aligned dumpaed: dump aed display dumpapa16: dump apa16 display	Mb/s Ethernet interface. Mc/s Mc/s Pethernet interface. Mc/s Mc/s Ethernet interface. Mc/s Ethernet interface. Mc/s M	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) mem(4) groups(1) mem(4) mem(4) melloc(3) malloc(3F) valloc(3C) dumpaed(1) dumpapa16(1)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group mem, kmem: main mem, kmem, kmem1, kmem2, kmem4, ros, afpamen: main malloc, free, realloc, calloc, alloca: malloc, free, felloc: valloc: aligned dumpaed: dump aed display dumpapa16: dump apa8 display dumpapa8: dump apa8 display	Mb/s Ethernet interface. Mcga Pel Display invert, rpow, msqrt, mcmp, move, Mcga Pel Display interface. Megabit ring. mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main mem, kmem: main memory. memberships. memory. memory. memory. memory allocator. memory allocator. memory as a binary file. memory as a binary file. memory as a binary file.	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) mem(4) groups(1) mem(4) mem(4) malloc(3) malloc(3F) valloc(3C) dumpapa16(1) dumpapa8(1)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group mem, kmem: main mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main malloc, free, realloc, calloc, alloca: malloc, free, felloc: valloc: aligned dumpaed: dump aed display dumpapa8: dump apa8 display dumpapa8: dump apa8 display dumpapa8c: dump apa8 display	Mb/s Ethernet interface. Mc/s Mc/s Pethernet interface. Mc/s Mc/s Ethernet interface. Mc/s Ethernet interface. Mc/s M	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) mem(4) groups(1) mem(4) mem(4) melloc(3) malloc(3F) valloc(3C) dumpaed(1) dumpapa16(1)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group mem, kmem: main mem, kmem, kmem1, kmem2, kmem4, ros, afpamen: main malloc, free, realloc, calloc, alloca: malloc, free, felloc: valloc: aligned dumpaed: dump aed display dumpapa16: dump apa8 display dumpapa8: dump apa8 display	Mb/s Ethernet interface. Mcga Pel Display invert, rpow, msqrt, mcmp, move, Mcga Pel Display interface. Megabit ring. mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main mem, kmem: main memory. memberships. memory. memory. memory. memory allocator. memory allocator. memory as a binary file. memory as a binary file. memory as a binary file.	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) mem(4) groups(1) mem(4) mem(4) malloc(3F) valloc(3F) valloc(3C) dumpapa16(1) dumpapa16(1) dumpapa8(1)
il: Interlan NI1010 10 ix: Interlan Np100 10 np: Interlan Np100 10 qe: DEC DEQNA Q-bus 10 /msub, mult, mdiv, pow, gcd, invert, rpow, msqrt, min, omin, fmin, m_in, mout,/ madd, msub, mult, ibm5081, mpel - IBM 5081 vv: Proteon proNET 10 memory. groups: show group mem, kmem: main mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main malloc, free, realloc, calloc, alloca: malloc, free, felloc: valloc: aligned dumpaed: dump aed display dumpapa8: dump apa8 display dumpapa8: dump apa8 display dumpapa8c: dump apa8 display	Mb/s Ethernet interface. Mcga Pel Display interface. Mega Pel Display interface. Megabit ring. mem, kmem, kmem1, kmem2, kmem4, ros, afpamem: main mem, kmem: main memory. memberships. memory. memory. memory. memory allocator. memory allocator. memory as a binary file.	en(4) ex(4) il(4) ix(4) np(4) qe(4) mp(3X) mp(3X) ibm5081(4) vv(4) n mem(4) groups(1) mem(4) mem(4) malloc(3) malloc(3F) valloc(3C) dumpapa16(1) dumpapa8(1) dumpapa8(1) dumpapa88(1) dumpapa8c(1)

_		
core: format of	memory image file.	core(5)
core: format of	memory image file	core(5)
vmstat: report virtual	memory statistics	vmstat(1)
vmstat: report virtual	memory statistics.	vmstat(1)
XMenu - X Deck of cards	Menu System.	XMenu(3X)
sort: sort or	merge files.	sort(1)
VI Merge: set	merge mode.	merge(3G)
omerge:	merge object files.	omerge(8)
. V .	• •	~ , ,
pmerge: pascal file	merger	pmerge(1)
	mesg: permit or deny messages	mesg(1)
mkstr: create an error	message file by massaging C source	mkstr(1)
recv, recvfrom, recvmsg: receive a	message from a socket	recv(2)
send, sendto, sendmsg: send a	message from a socket	send(2)
rvdgetm: get operations	message from Remote Virtual Disk (RVD) server	rvdgetm(8)
rvdsetm: set operations	message on Remote Virtual Disk (RVD) server	rvdsetm(8)
icmp: Internet Control	Message Protocol	icmp(4P)
error: analyze and disperse compiler error		error(1)
		7 7
error: analyze and disperse compiler error	messages	error(1)
mesg: permit or deny	messages	mesg(1)
syslogd: log systems	messages	syslogd(8)
syslogd: log systems	messages	syslogd(8)
perror, sys errlist, sys nerr: system error	messages	perror(3)
perror, gerror, ierrno: get system error	messages	perror(3F)
psignal, sys_siglist: system signal		*
		psignal(3)
msgs: system	messages and junk mail program.	msgs(1)
dmesg: collect system diagnostic	messages to form error log.	dmesg(8)
xmh: X window interface to the	mh Mail Handler	xmh(1)
invert, rpow, msqrt, mcmp, move, min, omin, fmin,	m in, mout, omout, fmout, m out, sdiv, itom://gcd, .	mp(3X)
/mdiv, pow, gcd, invert, rpow, msqrt, mcmp, move,	min, omin, fmin, m in, mout, omout, fmout, m out,/	mp(3X)
minidisk:	minidisk maintenance utility.	minidisk(8r)
THIN SOLL	minidisk: minidisk maintenance utility.	minidisk(8r)
	mkdir: make a directory.	mkdir(1)
	mkdir: make a directory file.	mkdir(2)
	mks: construct a file system	mkfs(8)
	mkhosts: generate hashed host table	mkhosts(8)
	mklost + found: make a lost + found directory for fsck	mklost + found(8)
	mknod: build special file.	mknod(8)
	mknod: make a special file.	mknod(2)
	mkpasswd: generate hashed password table	mkpasswd(8)
	mkproto: construct a prototype file system	mkproto(8)
source.	mkproto: construct a prototype file system mkstr: create an error message file by massaging C	mkproto(8) mkstr(1)
source.	mkproto: construct a prototype file system	mkproto(8)
source. chmod: change	mkproto: construct a prototype file system mkstr: create an error message file by massaging C	mkproto(8) mkstr(1)
	mkproto: construct a prototype file system mkstr: create an error message file by massaging C mktemp: make a unique file name	mkproto(8) mkstr(1) mktemp(3) chmod(1)
chmod: change getty: set terminal	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8)
chmod: change getty: set terminal VI_Merge: set merge	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode. mode. mode.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode of a file. mode of file.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode mask. mode of a file. mode of protocol line discipline.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode mask. mode of a file. mode of file. mode protocol line discipline. Mode X.25 IMP interface.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode mask. mode of a file. mode of lie. mode protocol line discipline.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, ldexp,	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode mask. mode of a file. mode of file. mode protocol line discipline. Mode X.25 IMP interface.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode of file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode of file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modified of a file. modified of a file. modified of a file. modifier utilities. module.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, ldexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup,	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile.	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup,	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151,	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode of file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Display interface.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm5151(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modifier utilities. module. module controller/drives. modules were used to construct a file. monocontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) ibm5151(4) ibm6153(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, mono: IBM 5151 ibm5153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor,	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modifier utilities. module. module controller/drives. modules were used to construct a file. monocontrol: prepare execution profile. mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. monostartup, moncontrol: prepare execution profile. monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. more, page: file perusal filter for crt viewing.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) monitor(3) monitor(3)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change ap: asynchronous data ddn: DDN Standard frexp, ldexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. monstartup, moncontrol: prepare execution profile. monochrome Graphics Display interface. monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. more, page: file perusal filter for crt viewing. motion.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) more(1) curses(3X)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor,	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. more, page: file perusal filter for crt viewing. motion. mount and dismount file system.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) monitor(3) monitor(3)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. more, page: file perusal filter for crt viewing. motion. mount and dismount file system.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) more(1) curses(3X)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor mount, umount:	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. montartup, moncontrol: prepare execution profile. montor, page: file perusal filter for crt viewing. mount and dismount file system. mount or remove file system.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6155(4) monitor(3) more(1) curses(3X) mount(8) mount(2)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor mount, umount:	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode mask. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modifier utilities. module. module controller/drives. modules were used to construct a file. monocontrol: prepare execution profile. monocontrol: prepare execution profile. monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monostartup, moncontrol: prepare execution profile. monochrome Graphics Display interface. monochrome Graphics Display interface. monostartup, moncontrol: prepare execution profile. more, page: file perusal filter for crt viewing. motion. mount and dismount file system. mount, umount: mount and dismount file system.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) more(1) curses(3X) mount(8) mount(2) mount(8)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor mount, umount:	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. module controller/drives. module. module swere used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. monstartup, moncontrol: prepare execution profile. monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. more, page: file perusal filter for crt viewing. motion. mount and dismount file system. mount or remove file system. mount, umount: mount and dismount file system. mount, umount: mount or remove file system.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) more(1) curses(3X) mount(8) mount(2)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor mount, umount: mount, umount:	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. monty, moncontrol: prepare execution profile. monty, moncontrol: prepare execution profile. more, page: file perusal filter for crt viewing. mouton. mount and dismount file system. mount, umount: mount and dismount file system. mount, umount: mount or remove file system. mount, umount: mount or remove file system. mount, umount: mount or remove file system.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) more(1) curses(3X) mount(8) mount(2) mount(8) mount(2) mtab(5)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor mount, umount: mount, umount: mount. X input emulator for queuing keyboard and	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. more, page: file perusal filter for crt viewing. motion. mount and dismount file system. mount, umount: mount or remove file system. mount, umount: mount or remove file system. mount, umount: mount or remove file system. mounted file system table. mouse events.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) more(1) curses(3X) mount(8) mount(2) mount(2) mtab(5) xemul(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor mount, umount: mount, umount:	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. monocontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. more, page: file perusal filter for crt viewing. motion. mount and dismount file system. mount, umount: mount and dismount file system. mount, umount: mount or remove file system. mount, umount: mount or remove file system. mounted file system table. mouse events. mouse interface.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) more(1) curses(3X) mount(8) mount(2) mount(2) mtab(5) xemul(4) mouse(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor mount, umount: mount, umount:	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. more, page: file perusal filter for crt viewing. motion. mount and dismount file system. mount or remove file system. mount, umount: mount and dismount file system. mount, umount: mount or remove file system. mount, umount: mount or remove file system. mounted file system table. mouse events. mouse interface. mouse: mouse interface.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) ibm5151(4) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) more(1) curses(3X) mount(8) mount(2) mtab(5) xemul(4) mouse(4) mouse(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor mount, umount: mount. mtab: xemul: X input emulator for queuing keyboard and mouse: /rpow, msqrt, mcmp, move, min, omin, fmin, m_in,	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. mont, page: file perusal filter for crt viewing. motion. mount and dismount file system. mount or remove file system. mount, umount: mount and dismount file system. mount, umount: mount or remove file system. mount, umount: mount or remove file system. mounted file system table. mouse events. mouse interface. mouse yell interface. mouse interface. mouse interface. mouse interface. mouse interface. mouse yell interface. mouse interface. mouse interface. mouse yell interface. mou	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) monitor(3) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) more(1) curses(3X) mount(8) mount(2) mount(2) mtab(5) xemul(4) mouse(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor mount, umount: mount, umount:	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. more, page: file perusal filter for crt viewing. motion. mount and dismount file system. mount or remove file system. mount, umount: mount and dismount file system. mount, umount: mount or remove file system. mount, umount: mount or remove file system. mounted file system table. mouse events. mouse interface. mouse: mouse interface.	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) ibm5151(4) ibm5151(4) ibm6153(4) ibm6155(4) monitor(3) more(1) curses(3X) mount(8) mount(2) mtab(5) xemul(4) mouse(4) mouse(4)
chmod: change getty: set terminal VI_Merge: set merge umask: set file creation chmod: change chmod, fchmod: change ap: asynchronous data ddn: DDN Standard frexp, Idexp, touch: update date last xmodmap, xprkbd - X Window System keyboard rmt: remote magtape protocol up: unibus storage what: show what versions of object monitor, monstartup, profile. ibm5151, ibm5151, mono: IBM 5151 ibm6153, apa8: IBM 6153 Advanced ibm6155, apa16: IBM 6155 Extended monitor, curses: screen functions with "optimal" cursor mount, umount: mount. mtab: xemul: X input emulator for queuing keyboard and mouse: /rpow, msqrt, mcmp, move, min, omin, fmin, m_in,	mkproto: construct a prototype file system. mkstr: create an error message file by massaging C mktemp: make a unique file name. mode. mode. mode. mode of a file. mode of a file. mode protocol line discipline. Mode X.25 IMP interface. modf: split into mantissa and exponent. modified of a file. modifier utilities. module. module controller/drives. modules were used to construct a file. moncontrol: prepare execution profile. monitor, monstartup, moncontrol: prepare execution mono: IBM 5151 Monochrome Display interface. Monochrome Graphics Display interface. Monochrome Graphics Display interface. monstartup, moncontrol: prepare execution profile. mont, page: file perusal filter for crt viewing. motion. mount and dismount file system. mount or remove file system. mount, umount: mount and dismount file system. mount, umount: mount or remove file system. mount, umount: mount or remove file system. mounted file system table. mouse events. mouse interface. mouse yell interface. mouse interface. mouse interface. mouse interface. mouse interface. mouse yell interface. mouse interface. mouse interface. mouse yell interface. mou	mkproto(8) mkstr(1) mktemp(3) chmod(1) getty(8) merge(3G) umask(2) chmod(3F) chmod(2) ap(4) ddn(4) frexp(3) touch(1) xmodmap(1) rmt(8C) up(4) what(1) monitor(3) ibm5151(4) ibm6153(4) ibm6153(4) ibm6155(4) monitor(3) more(1) curses(3X) mount(8) mount(2) mount(8) mount(2) mtab(5) xemul(4) mouse(4) mp(3X)

/mult, mdiv, pow, gcd, invert, rpow, msqrt, mcmp,		
/man, mar, pow, god, mvere, rpow, magre, momp,	move, min, omin, fmin, m_in, mout, omout, fmout,/	mp(3X)
mv:	move or rename files	mv(1)
lseek:	move read/write pointer	lseek(2)
		· /
VI_AMove, VI_RMove:	move the current point	
hk: RK6-11/RK06 and RK07	moving head disk	hk(4)
ibm5081,	mpel - IBM 5081 Mega Pel Display interface	
101110001,	mset: retrieve ASCII to IBM 3270 keyboard map	
		mset(1)
	mset: retrieve ASCII to IBM 3270 keyboard map	mset(1)
	msgs: system messages and junk mail program	msgs(1)
madd, msub, mult, mdiv, pow, gcd, invert, rpow,	msqrt, mcmp, move, min, omin, fmin, m in, mout,/	mp(3X)
mcmp, move, min, omin, fmin, m_in, mout,/ madd,	msub, mult, mdiv, pow, gcd, invert, rpow, msqrt,	mp(3X)
	mt: magnetic tape manipulating program	mt(1)
	mt: magnetic tape manipulating program	
	mt: TM78/TU-78 MASSBUS magtape interface	
	mtab: mounted file system table	mtab(5)
¥	mtio: 4.3/RT magtape interface	
	mtio: UNIX magtape interface	
move, min, omin, fmin, m in, mout,/ madd, msub,	mult, mdiv, pow, gcd, invert, rpow, msqrt, mcmp,	mp(3X)
fmin, m in, mout, omout, fmout, m out, sdiv, itom:	multiple precision integer arithmetic. /min, omin,	* i===i
dh: DH-11/DM-11 communications	multiplexer	dh(4)
dhu: DHU-11 communications	multiplexer	dhu(4)
dz: DZ-11 communications	multiplexer	dz(4)
select: synchronous I/O	multiplexing	
dmf: DMF-32, terminal	multiplexor	dmf(4)
dmz: DMZ-32 terminal	multiplexor	dmz(4)
interface. asy:	multi-port asynchronous communications RS232C	asy(4)
fsplit: split a	multi-routine Fortran file into individual files	fsplit(1)
switch:	multi-way command branch	csh(1)
	mv: move or rename files	mv(1)
C		
from: who is	my mail from?.	from(1)
getsockname: get socket	name	getsockname(2)
pwd: working directory	name.	pwd(1)
		•
tty: get terminal	name.	tty(1)
getdiskbyname: get disk description by its	name	getdisk(3)
getenv: value for environment	name.	getenv(3)
getlog: get user's login	name.	getlog(3F)
getlogin: get login	name	getlogin(3)
mktemp: make a unique file	name	mktemp(3)
• •		
hosts: host	name data base.	hosts(5)
networks: network	name data base	networks(5)
protocols: protocol	name data base	protocols(5)
services: service	name data base.	services(5)
whois: DARPA Internet user	name directory service.	whois(1)
getpw: get	name from uid	getpw(3C)
nm: print	name list.	nm(1)
•		
symorder: rearrange	name list	symorder(1)
		nlist(3)
nlist: get entries from	name list	111131(3)
rename: change the	name of a file	rename(2)
rename: change the ttyname, isatty, ttyslot: find	name of a file	rename(2) ttyname(3)
rename: change the	name of a file	rename(2)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find	name of a file	rename(2) ttyname(3) ttynam(3F)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get	name of a file	rename(2) ttyname(3) ttynam(3F) getpeername(2)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set	name of a file	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get	name of a file	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set	name of a file	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get	name of a file	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get hostnm: get	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server.	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get hostname: set or print	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket.	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostname hostname: set or print named: Internet domain bind: bind a	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server.	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get hostnm: get	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket.	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostname: set or print named: Internet domain bind: bind a	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names.	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers.	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostname: set or print named: Internet domain bind: bind a	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs.	rename(2) ttyname(3) ttyname(3F) getpeername(2) gethostname(2) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers.	rename(2) ttyname(3) ttyname(3) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs.	rename(2) ttyname(3) ttyname(3) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. necheck: generate names from i-numbers. neqn, checkeq: typeset mathematics.	rename(2) ttyname(3) ttyname(3) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host system. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network.	rename(2) ttyname(3) ttyname(3) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore.	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. netstat: show network status.	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostname: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore.	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. netstat: show network status.	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore.	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. netstat: show network status.	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostname: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. net; install system from tape or over network. netstat: show network status. network. network. network. restore.tape,	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostname: get/set hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. netstat: show network status. network. network. network.	rename(2) ttyname(3) ttyname(3) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore(8C)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty; find getpeername: get gethostname, sethostname: get/set hostnm: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the ntohl, ntohs: convert values between host and	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. network. network. network. network. network byte order. htonl, htons,	rename(2) ttyname(3) ttyname(3) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore.tape(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostname: get/set hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. neheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. network. network. network. network byte order. htonl, htons, network entry. getnetent, getnetbyaddr,	rename(2) ttyname(3) ttyname(3) ttyname(3) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore.tape(8) getpeerder(3N) getnetent(3N)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty; find getpeername: get gethostname, sethostname: get/set hostnm: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the ntohl, ntohs: convert values between host and getnetbyname, setnetent, endnetent: get	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. neheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. network. network. network. network byte order. htonl, htons, network entry. getnetent, getnetbyaddr,	rename(2) ttyname(3) ttyname(3) ttyname(3) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore.tape(8) getpeerder(3N) getnetent(3N)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the ntohl, ntohs: convert values between host and getnetbyname, setnetent, endnetent: get gethostent, sethostent, endhostent: get	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. network. network. network. network byte order. htonl, htons, network entry. getnetent, getnetbyaddr, network host entry. gethostbyname, gethostbyaddr,	rename(2) ttyname(3) ttyname(3) ttyname(3) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore.tape(8) restore.tape(8) yetpeorder(3N) getnetent(3N) gethostbyname(3N)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty; find getpeername: get gethostname, sethostname: get/set hostname: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the ntohl, ntohs: convert values between host and getnetbyname, setnetent, endnetent: get gethostent, sethostent, endhostent: get ping: send ICMP ECHO_REQUEST packets to	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. network. network. network restore.tape, network byte order. htonl, htons, network bost entry. getnetent, getnetbyaddr, network host entry. gethostbyname, gethostbyaddr, network hosts.	rename(2) ttyname(3) ttyname(3) ttyname(3) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore(8C) byteorder(3N) getnetent(3N) gethostbyname(3N) ping(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty: find getpeername: get gethostname, sethostname: get/set hostnm: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the ntohl, ntohs: convert values between host and getnetbyname, setnetent, endnetent: get gethostent, sethostent, endhostent: get	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. network. network. network. network byte order. htonl, htons, network entry. getnetent, getnetbyaddr, network host entry. gethostbyname, gethostbyaddr,	rename(2) ttyname(3) ttyname(3) ttyname(3) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore.tape(8) prestore.tape(8) netstat(1) rdump(8C) restore.tape(8) restore.tape(8) restore.tape(8)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty; find getpeername: get gethostname, sethostname: get/set hostname: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the ntohl, ntohs: convert values between host and getnetbyname, setnetent, endnetent: get gethostent, sethostent, endhostent: get ping: send ICMP ECHO_REQUEST packets to imp: 1822	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. network. network. network. network restore.tape, network byte order. htonl, htons, network byte order. htonl, htons, network host entry. getnestbyaddr, network hosts. network hosts. network interface.	rename(2) ttyname(3) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore(8C) byteorder(3N) getnestent(3N) getnostbyname(3N) ping(8) imp(4)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty; find getpeername: get gethostname, sethostname: get/set hostname: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the ntohl, ntohs: convert values between host and getnetbyname, setnetent, endnetent: get gethostent, sethostent, endhostent: get ping: send ICMP ECHO_REQUEST packets to imp: 1822 lo: software loopback	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. network. network. network restore.tape, network byte order. htonl, htons, network host entry. gethostbyname, gethostbyaddr, network hosts. network interface.	rename(2) ttyname(3) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore(8C) byteorder(3N) gethostbyname(3N) ping(8) imp(4) lo(4)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty; find getpeername: get gethostname, sethostname: get/set hostname: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the ntohl, ntohs: convert values between host and getnetbyname, setnetent, endhostent: get gethostent, sethostent, endhostent: get ping: send ICMP ECHO_REQUEST packets to imp: 1822 lo: software loopback pcl: DEC CSS PCL-11 B	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. net, checkeq: typeset mathematics. net: install system from tape or over network. netstat: show network status. network. network. network restore.tape, network host entry. getnetent, getnetbyaddr, network hosts. network interface. network interface. Network Interface.	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore(8C) byteorder(3N) gethostbyname(3N) ping(8) imp(4) lo(4) pcl(4)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty; find getpeername: get gethostname, sethostname: get/set hostname: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the ntohl, ntohs: convert values between host and getnetbyname, setnetent, endnetent: get gethostent, sethostent, endhostent: get ping: send ICMP ECHO_REQUEST packets to imp: 1822 lo: software loopback	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. neqn, checkeq: typeset mathematics. net: install system from tape or over network. network. network. network restore.tape, network byte order. htonl, htons, network host entry. gethostbyname, gethostbyaddr, network hosts. network interface.	rename(2) ttyname(3) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore(8C) byteorder(3N) gethostbyname(3N) ping(8) imp(4) lo(4)
rename: change the ttyname, isatty, ttyslot: find ttynam, isatty; find getpeername: get gethostname, sethostname: get/set hostname: get hostname: set or print named: Internet domain bind: bind a foreach: loop over list of ncheck: generate rvdexch: exchange eqn, restore.tape, restore. rdump: file system dump across the restore.net: install system from tape or over rrestore: restore a file system dump across the ntohl, ntohs: convert values between host and getnetbyname, setnetent, endhostent: get gethostent, sethostent, endhostent: get ping: send ICMP ECHO_REQUEST packets to imp: 1822 lo: software loopback pcl: DEC CSS PCL-11 B	name of a file. name of a terminal. name of a terminal port. name of connected peer. name of current host. name of current host. name of current host. name of current host system. name server. name to a socket. named: Internet domain name server. names. names from i-numbers. names of two Remote Virtual Disk (RVD) packs. ncheck: generate names from i-numbers. net, checkeq: typeset mathematics. net: install system from tape or over network. netstat: show network status. network. network. network restore.tape, network host entry. getnetent, getnetbyaddr, network hosts. network interface. network interface. Network Interface.	rename(2) ttyname(3) ttynam(3F) getpeername(2) gethostname(2) hostnm(3F) hostname(1) named(8) bind(2) named(8) csh(1) ncheck(8) rvdexch(8) ncheck(8) eqn(1) restore.tape(8) netstat(1) rdump(8C) restore.tape(8) rrestore(8C) byteorder(3N) gethostbyname(3N) ping(8) imp(4) lo(4) pcl(4)

ifconfig: configure	network interface parameters	ifconfig(8c)
slattach: attach serial lines as	network interfaces	slattach(8C)
networks:	network name data base	networks(5)
routed:	network routing daemon.	routed(8C)
netstat: show hy:	network status	
ns: Xerox	Network Systems (tm) protocol family	
networking: introduction to	networking facilities	
	networking: introduction to networking facilities	
	networks: network name data base	
creat: create a	new file.	creat(2)
open a file for reading or writing, or create a	new file. open:	open(2)
newfs: construct a	new file system.	` '
newfs: construct a	new file system.	*. *
newvd: create a fork: create a	new filesystem on a Remote Virtual Disk (RVD) new process	` '
vfork: spawn	new process in a virtual memory efficient way.	
login: login	new user.	
adduser: procedure for adding	new users.	
aliases file.	newaliases: rebuild the data base for the mail	
	newfs: construct a new file system	
	newfs: construct a new file system	
Disk (RVD).	newvd: create a new filesystem on a Remote Virtual	· ·
/classfloat, isnan, unordered, finite, infinity,	nextdouble, nextfloat, fptestround, fpsetround,/	
/isnan, unordered, finite, infinity, nextdouble, dbminit, fetch, store, delete, firstkey,	nextfloat, fptestround, fpsetround, swapround,/ nextkey: data base subroutines	ieee(3) dbm(3X)
dommit, fetch, store, delete, mistkey,	nfabort: dump core and log it in a notesfile	
system.	nfcomment: a user interface to the notesfile	nfcomment(3)
il: Interlan	NI1010 10 Mb/s Ethernet interface	il(4)
gettable: get	NIC format host tables from a host	
htable: convert	NIC standard format host tables	htable(8)
vgrind: grind	nice listings of programs.	vgrind(1)
Pageprinter. vgrind: grind	nice listings of programs for the IBM 3812	
(sh only).	nice, nohup: run a command at low priority	nice(1)
	nice: run low priority process	csh(1) nice(3C)
	nlist: get entries from name list.	
	nm: print name list.	nm(1)
only). nice,	nohup: run a command at low priority (sh	
	nohup: run command immune to hangups	csh(1)
setjmp, longjmp:	non-local goto.	setjmp(3)
bit: and, or, xor,	not, rshift, Ishift bitwise functions	bit(3F)
nfabort: dump core and log it in a	notesfile	nfabort(3)
nfcomment: a user interface to the	notesfile system.	nfcomment(3)
notify: request immediate biff: be	notification	csh(1) biff(1)
om. oc	notify: request immediate notification	
	np: Interlan Np100 10 Mb/s Ethernet interface	
ix: Interlan	Np100 10 Mb/s Ethernet interface	ix(4)
np: Interlan	Np100 10 Mb/s Ethernet interface	np(4)
Graphics Printer. proff:	nroff for the IBM 4201 Proprinter and IBM 5152	proff(1)
soelim: eliminate .so's from	nroff input.	soelim(1)
tbl: format tables for	nroff or troff.	tbl(1)
colort: filter prfl: IBM 4201 Proprinter/IBM 5152 Graphics Printer	nroff output for CRT previewing	colcrt(1)
prii. 16 ivi 4201 Proprinter/16 ivi 3132 Graphics Printer	nroff post-processing filter	
troff,	nroff: text formatting and typesetting	
deroff: remove	nroff, troff, tbl and eqn constructs	
vlp: Format Lisp programs to be printed with	nroff, vtroff, or troff	
checknr: check	nroff/troff files	checknr(1)
nsip: software network interface encapsulating	ns packets in ip packets	• • •
XNSrouted:	NS Routing Information Protocol daemon	XNSrouted(8C)
routines.	ns: Xerox Network Systems(tm) protocol family ns addr, ns ntoa: Xerox NS(tm) address conversion	ns(4F) ns(3N)
packets in ip packets	nsip: software network interface encapsulating ns	nsip(4)
ns addr,	ns ntoa: Xerox NS(tm) address conversion routines	ns(3N)
ns_addr, ns_ntoa: Xerox	NS(tm) address conversion routines	ns(3N)
network byte order. htonl, htons,	ntohl, ntohs: convert values between host and	byteorder(3N)
order. htonl, htons, ntohl,	ntohs: convert values between host and network byte .	byteorder(3N)
	null: data sink.	nuli(4)
phones: remote host phone rand, srand: random	number data base	phones(5)
random, drandm, irandm: better random	number generator	rand(3C) random(3F)
random, srandom, initstate, setstate: better random	number generator; routines for changing generators.	
L-dialcodes: UUCP phone	number index file	
intro: introduction to system calls and error	numbers	intro(2)

	•	(0)
intro: introduction to system calls and error	numbers	intro(2)
atof, atoi, atol: convert ASCII to	numbers	atof(3)
idate, itime: return date or time in	numerical form.	idate(3F)
loc: return the address of an	object	loc(3F)
long, short: integer	object conversion.	long(3F)
size: size of an	object file.	size(1)
omerge: merge	object files.	omerge(8)
lorder: find ordering relation for an	object library.	lorder(1)
what: show what versions of	object modules were used to construct a file	
		what(1)
strings: find the printable strings in a	object, or other binary, file	strings(1)
index, rindex, lnblnk, len: tell about character	objects	index(3F)
line discipline for machine-machine communication	(obsolete). bk:	bk(4)
od:	octal, decimal, hex, ascii dump	od(1)
	od: octal, decimal, hex, ascii dump	od(1)
	omerge: merge object files.	omerge(8)
Inorre and insert many waget many many min		
/pow, gcd, invert, rpow, msqrt, mcmp, move, min,	omin, fmin, m_in, mout, omout, fmout, m_out, sdiv,/	
/msqrt, mcmp, move, min, omin, fmin, m_in, mout,	omout, fmout, m_out, sdiv, itom: multiple precision/	
rvdcopy: copy contents of	one RVD disk pack to another	rvdcopy(8)
	onintr: process interrupts in command scripts	csh(1)
afpacode: load, test, and bring	online the Advanced Floating Point Accelerator	afpacode(8r)
nohup: run a command at low priority (sh	only). nice,	nice(1)
program file including aliases and paths (csh	only). which: locate a	which(1)
file. open:	open a file for reading or writing, or create a new	open(2)
fopen, freopen, fdopen:	open a stream.	fopen(3S)
flock: apply or remove an advisory lock on an	open file.	flock(2)
a new file.	open: open a file for reading or writing, or create	open(2)
closedir: directory operations.	opendir, readdir, telldir, seekdir, rewinddir,	directory(3)
syslog,	openlog, closelog, setlogmask: control system log	syslog(3)
cont, point, linemod, space, closepl:/ plot:	openpl, erase, label, line, circle, arc, move,	plot(3X)
(3X) libraries plot:	openpl et al.: f77 library interface to plot	plot(3F)
savecore: save a core dump of the	operating system.	savecore(8)
kgmon: generate a dump of the	operating system's profile buffers	kgmon(8)
intro: introduction to system maintenance and	operation commands	intro(8)
intro: introduction to system maintenance and	operation commands	intro(8)
tgetstr, tgoto, tputs: terminal independent	operation routines. tgetent, tgetnum, tgetflag,	* *
		termcap(3X)
bcopy, bcmp, bzero, ffs: bit and byte string	operations	bstring(3)
telldir, seekdir, rewinddir, closedir: directory	operations. opendir, readdir,	directory(3)
strcpy, strncpy, strlen, index, rindex: string	operations. streat, streat, stremp, strnemp,	string(3)
server. rvdgetm: get	operations message from Remote Virtual Disk (RVD)	rvdgetm(8)
server. rvdsetm: set	operations message on Remote Virtual Disk (RVD)	r vdsetm(8)
infnan: signals invalid floating-point	operations on a VAX (temporary)	infnan(3M)
	operator	
join: relational database	operator.	join(1)
curses: screen functions with	"optimal" cursor motion	curses(3X)
getopt: get	option letter from argv	getopt(3)
stty: set terminal	options.	
getsockopt, setsockopt: get and set	options on sockets.	getsockopt(2)
lastcomm: show last commands executed in reverse	order	lastcomm(1)
ntohs: convert values between host and network byte	order. htonl, htons, ntohl,	
lorder: find	ordering relation for an object library	• , ,
		· ·
bessel functions: of two kinds for integer	orders	bessel(3F)
VI_Force: force output of graphics	orders	force(3G)
vi: screen	oriented (visual) display editor based on ex	vi(1)
a.out: assembler and link editor	output	a.out(5)
a.out: assembler and link editor	output.	a.out(5)
terminate a process after flushing any pending	output. exit:	exit(3)
write, writev: write	output.	write(2)
ecvt, fcvt, gcvt:	output conversion.	ecvt(3)
	•	* . *
printf, fprintf, sprintf: formatted	output conversion.	printf(3S)
fold: fold long lines for finite width	output device.	fold(1)
stdemul: standard	output emulator.	stdemul(4)
5152 Graphics Printer. ibmbit, ibmgra, ibmpro:	output filters for the IBM 4201 Proprinter and IBM .	lpfilter(8r)
colort: filter nroff	output for CRT previewing	colcrt(1)
VI Force: force	output of graphics orders	force(3G)
flush: flush	output to a logical unit.	flush(3F)
Xtext: routines to provide simple text	output windows.	xtext(3X)
foreach: loop	over list of names.	csh(1)
the control of the co		
restore.net: install system from tape or	over network. restore.tape,	restore.tape(8)
sendmail: send mail	over the internet.	sendmail(8)
trapov: trap and repair floating point	overflow	trapov(3F)
exec:	overlay shell with specified command	csh(1)
chown: change	owner	chown(8)
chown, fchown: change	owner and group of a file.	chown(2)
quot: summarize file system	ownership.	quot(8)
quot. summanze me system		uuvuo:
f 1 6 m	pac: printer/plotter accounting information	pac(8)
force spindown of a Remote Virtual Disk (RVD)	pac: printer/plotter accounting information pack. rvddown:	pac(8) rvddown(8)
force spindown of a Remote Virtual Disk (RVD) spindown: spin up/down Remote Virtual Disk (RVD)	pac: printer/plotter accounting information	pac(8)

rvdcopy: copy contents of one RVD disk	pack to another.	rvdcopy(8)
spp: Xerox Sequenced	Packet Protocol.	spp(4P)
trsp: transliterate sequenced	packet protocol trace	trsp(8C)
network interface encapsulating ns packets in ip	packets. nsip: software	nsip(4) nsip(4)
nsip: software network interface encapsulating ns ping: send ICMP ECHO_REQUEST	packets in ip packets	ping(8)
format: how to format disk	packs.	format(8V)
exchange names of two Remote Virtual Disk (RVD)	packs. rvdexch:	rvdexch(8)
spindown client's Remote Virtual Disk (RVD)	packs. rvdflush:	rvdflush(8)
back up and restore Remote Virtual Disk (RVD)	packs to and from tape. savervd, zaprvd, savephys:	savervd(8)
more,	page: file perusal filter for crt viewing	more(1)
getpagesize: get system	page size	getpagesize(2)
pagesize: print system	page size.	pagesize(1)
3820 and IBM 3800 fonts for use with the IBM 3812	Pageprinter. /cvt20to12, cvt00to12: convert IBM	cvt3812(8)
pprint: print text files on IBM 3812	Pageprinter	pprint(1)
ppt: spooling system filter for the IBM 3812	Pageprinter	ppt(8)
ptroff: print troff files on IBM 3812	Pageprinter	ptroff(1)
grind nice listings of programs for the IBM 3812	Pageprinter vgrind:	vgrind(1)
width3812: build width tables for IBM 3812	Pageprinter fonts	width3812(8)
ibm3812pp: IBM 3812	Pageprinter server.	ibm3812pp(8)
printer3812: IBM 3812	Pageprinter status information.	printer3812(5)
· ·	pagesize: print system page size	pagesize(1)
tk:	paginator for the Tektronix 4014	tk(1)
swapon: specify additional device for drum:	paging and swapping	swapon(8) drum(4)
swapon: add a swap device for interleaved	paging device.	swapon(2)
swapon. add a swap device for interleaved socketpair: create a	pair of connected sockets.	socketpair(2)
xsetroot: X window system root window	parameter setting utility.	xsetroot(1)
ifconfig: configure network interface	parameters.	ifconfig(8C)
ifconfig: configure network interface	parameters.	ifconfig(8c)
VI_QMerge, VI_QPoint, VI_QWidth: query graphics	parameters. /VI_QColor, VI_QDash, VI_QFont,	query(3G)
diskpart: calculate default disk	partition sizes.	diskpart(8)
diskpart: calculate default disk	partition sizes	diskpart(8)
fdisk: boot record	partition table maintenance utility	fdisk(8)
pc:	Pascal compiler	pc(1)
pp: Professional	Pascal compiler	pp(1)
pxref:	Pascal cross-reference program	pxref(1)
pdx:	pascal debugger	pdx(1)
pxp:	Pascal execution profiler	pxp(1)
pmerge:	pascal file merger	pmerge(1)
px:	Pascal interpreter	px(1)
pix:	Pascal interpreter and executor	pix(1)
pi:	Pascal interpreter code translator	pi(1)
chfn, chsh,	passwd: change password file information.	passwd(1)
loss tell Venue about vous	passwd: password file.	passwd(5)
log: tell Venus about your getpass: read a	password	log(1) getpass(3)
passwd:	password file.	passwd(5)
•	password file.	vipw(8)
	password file entry. getpwent, getpwuid,	getpwent(3)
chfn, chsh, passwd: change	password file information.	passwd(1)
mkpasswd: generate hashed	password table.	mkpasswd(8)
getwd: get current working directory	pathname.	getwd(3)
getcwd: get	pathname of current working directory	getcwd(3F)
USERFILE: UUCP	pathname permissions file.	USERFILE(5)
which: locate a program file including aliases and	paths (csh only)	which(1)
grep, egrep, fgrep: search a file for a	pattern	grep(1)
lptest: generate lineprinter ripple	pattern.	lptest(1)
VI_Dash: set line dash	pattern	dash(3G)
awk:	pattern scanning and processing language.	awk(1)
	pause: stop until signal	pause(3C)
debug: debugger for the IBM RT	PC	debug(8)
kbdlock: lock the keyboard of the IBM RT	PC	kbdlock(1)
un: IBM RT	PC Baseband Adapter for use with Ethernet	un(4)
	pc: Pascal compiler	pc(1)
lan: IBM RT	PC Token-Ring Adapter	lan(4)
	pcc: pcc-based C compiler	pcc(1)
pcc;	pcc-based C compiler	pcc(1)
dosread: read, write, dir, delete on	PC-DOS diskette	dosread(1)
-1. DEC 000	pcl: DEC CSS PCL-11 B Network Interface	pcl(4)
pcl: DEC CSS	PCL-11 B Network Interface	pcl(4)
popen,	ncinca initiata I/() to/from a process	popen(3)
	pclose: initiate I/O to/from a process	
getneername, get name of connected	pdx: pascal debugger	pdx(1)
getpeername: get name of connected	pdx: pascal debugger	pdx(1) getpeername(2)
getpeername: get name of connected ibm5081, mpel – IBM 5081 Mega exit: terminate a process after flushing any	pdx: pascal debugger	pdx(1)

L.cmds: UUCP remote command		Lemds(5)
USERFILE: UUCP pathname mesg:	permissions file	USERFILE(5) mesg(1)
ptx:	permuted index	O. ,
limit: alter	per-process resource limitations	csh(1)
	perror, gerror, ierrno: get system error messages	
messages.	perror, sys_errlist, sys_nerr: system error	
sticky: landump: dump IBM Token-Ring	persistent text and append-only directories Personal Computer Adapter	
more, page: file	perusal filter for crt viewing.	more(1)
more, page. me	pf: set keyboard program-function keys.	
phones: remote host	phone number data base	phones(5)
L-dialcodes: UUCP	phone number index file	
	phones: remote host phone number data base	
ct:	phototypesetter interface	
tc:	photoypesetter simulator	tc(1) pi(1)
pictures.	pic: troff preprocessor for drawing simple	pic(1)
ps: Evans and Sutherland	Picture System 2 graphics device interface	ps(4)
pic: troff preprocessor for drawing simple	pictures	pic(1)
hosts.	ping: send ICMP ECHO_REQUEST packets to network	ping(8)
	pipe: create an interprocess communication channel	pipe(2)
tee:	pipe fitting	tee(1)
bg:	place job in background.	pix(1) csh(1)
psp:	planar serial port RS232C interface	psp(4)
r-r-	plot: graphics filters	plot(1G)
	plot: graphics interface	plot(5)
move, cont, point, linemod, space, closepl:/	plot: openpl, erase, label, line, circle, arc,	
plot (3X) libraries	plot: openpl et al.: f77 library interface to	plot(3F)
VI AMove, VI RMove: move the current	pmerge: pascal file merger	pmerge(1) move(3G)
load, test, and bring online the Advanced Floating	Point Accelerator. afpacode:	afpacode(8r)
fpa: direct interface to floating	point accelerator	fpa(3X)
trpfpe, fpecnt: trap and repair floating	point faults.	trpfpe(3F)
/erase, label, line, circle, arc, move, cont,	point, linemod, space, closepl: graphics interface	plot(3X)
trapov: trap and repair floating getfloatstate: return machine and process floating	point overflow	trapov(3F) getfloatstate(2)
lseek: move read/write	pointer	lseek(2)
dmc: DEC DMC-11/DMR-11	point-to-point communications device.	dmc(4)
uupoll:	poll a remote UUCP site	uupoll(8C)
popd:	pop shell directory stack	csh(1)
	popd: pop shell directory stack.	csh(1)
ttymam ignttyy find name of a tarminal	popen, pclose: initiate I/O to/from a process port	popen(3)
ttynam, isatty: find name of a terminal psp: planar serial	port RS232C interface	ttynam(3F) psp(4)
IBM 4201 Proprinter/IBM 5152 Graphics Printer nroff	post-processing filter. prfl:	
exp, expm1, log, log10, log1p,	pow: exponential, logarithm, power	exp(3M)
omin, fmin, m_in, mout,/ madd, msub, mult, mdiv,	pow, gcd, invert, rpow, msqrt, mcmp, move, min,	mp(3X)
log, log10, log1p, pow: exponential, logarithm,	power. exp, expm1,	
.PP uwm - Window Manager Client Application of X.	PP.	uwm(1)
.PP	pp. Professional Pascal compiler	pp(1) uwm(1)
	pprint: print text files on IBM 3812 Pageprinter	pprint(1)
Pageprinter.	ppt: spooling system filter for the IBM 3812	ppt(8)
	pr: print file.	pr(1)
mout, omout, fmout, m_out, sdiv, itom: multiple	precision integer arithmetic. /omin, fmin, m_in,	mp(3X)
monitor, monstartup, moncontrol: pic: troff	prepare execution profile	monitor(3) pic(1)
colort: filter proff output for CRT	previewing.	colcrt(1)
nroff post-processing filter.	prfl: IBM 4201 Proprinter/IBM 5152 Graphics Printer .	prfl(1)
types:	primitive system data types	types(5)
cat: catenate and	print	cat(1)
Ipr: off line	print.	lpr(1)
rvdhosts: date:	print a list of RVD servers	rvdhosts(8) date(1)
date:	print and set the date.	date(1)
cal:	print calendar	cal(1)
hashstat:	print command hashing statistics.	csh(1)
jobs:	print current job list	csh(1)
whoami: nr:	print effective current user id	whoami(1) pr(1)
fpr:	print Fortran file	fpr(1)
history:	print history event list	csh(1)
hostid: set or	print identifier of current host system	hostid(1)
bitprt: capture the image on a bitmap display and	print it on an IBM printer	bitprt(1)

nm	print name list.	nm(1)
nm: hostname: set or	• • • • • • • • • • • • • • • • • • • •	nm(1) hostname(1)
printenv:	print name of current host system	printenv(1)
man: find manual information by keywords;	print out the manual	man(1)
pstat:	print system facts.	pstat(8)
pstat:	print system facts.	pstat(8)
pagesize:	print system page size	pagesize(1)
pprint:	print text files on IBM 3812 Pageprinter	pprint(1)
atq:	print the queue of jobs waiting to be run	atq(1)
ptroff;	print troff files on IBM 3812 Pageprinter	ptroff(1)
diction, explain:	print wordy sentences; thesaurus for diction	diction(1)
xpr:	print X window dump	xpr(1)
file. strings: find the	printable strings in a object, or other binary,	strings(1)
	printcap: printer capability data base	printcap(5)
vlp: Format Lisp programs to be	printed with nroff, vtroff, or troff	vlp(1)
	printenv: print out the environment	printenv(1)
image on a bitmap display and print it on an IBM	printer. bitprt: capture the	bitprt(1)
lp: line	printer	lp(4)
lp: line	printer	lp(4)
for the IBM 4201 Proprinter and IBM 5152 Graphics	Printer. ibmbit, ibmgra, ibmpro: output filters	lpfilter(8r)
for the IBM 4201 Proprinter and IBM 5152 Graphics	Printer. proff: nroff	proff(1)
printcap:	printer capability data base	printcap(5)
lpc: line	printer control program	lpc(8)
lpd: line	printer daemon	lpd(8)
prfl: IBM 4201 Proprinter/IBM 5152 Graphics	Printer nroff post-processing filter	prfl(1)
lprm: remove jobs from the line	printer spooling queue.	lprm(1)
information.	printer3812: IBM 3812 Pageprinter status	printer3812(5)
pac:	printer/plotter accounting information	pac(8)
conversion.	printf, fprintf, sprintf: formatted output	printf(3S)
setpriority: get/set program scheduling	priority, getpriority,	getpriority(2)
nice: set program	priority.	nice(3C)
nice, nohup: run a command at low	priority (sh only)	nice(1)
renice: alter	priority of running processes	renice(8)
nice: run low	priority process.	csh(1)
adduser:	procedure for adding new users	adduser(8)
reboot: UNIX bootstrapping	procedures	reboot(8)
reboot: bootstrapping	procedures	reboot(8)
nice: run low priority	process.	csh(1)
stop: halt a job or	process.	csh(1)
_exit: terminate a	process	exit(2)
fork: create a new	process	fork(2)
implogd: IMP logger	process	implogd(8C)
kill: send signal to a	process.	kill(2)
fork: create a copy of this	process.	fork(3F)
kill: send a signal to a popen, pclose: initiate I/O to/from a	process.	kill(3F)
wait: await completion of	process	popen(3)
VI Run:	process a log file.	wait(1) run(3G)
exit: terminate a	process after flushing any pending output.	exit(3)
init:	process control initialization.	init(8)
init:	process control initialization.	init(8)
getfloatstate: return machine and	process floating point state.	getfloatstate(2)
getpgrp: get	process group.	getpgrp(2)
killpg: send signal to a	process group.	killpg(2)
setpgrp: set	process group.	setpgrp(2)
getpid: get	process id.	getpid(3F)
getpid, getppid: get	process identification.	getpid(2)
vfork: spawn new	process in a virtual memory efficient way	vfork(2)
onintr:	process interrupts in command scripts	csh(1)
ps:	process status.	ps(1)
times: get	process times.	times(3C)
wait: wait for a	process to terminate	wait(3F)
wait, wait3: wait for	process to terminate	wait(2)
ptrace:	process trace.	ptrace(2)
ptrace:	process trace.	ptrace(2)
kill: terminate a	process with extreme prejudice.	kill(1)
exit: terminate	process with status.	exit(3F)
kill: kill jobs and	processes	csh(1)
gcore: get core images of running	processes	gcore(1)
renice: alter priority of running	processes	renice(8)
wait: wait for background	processes to complete	csh(1)
awk: pattern scanning and	processing language	awk(1)
halt: stop the	processor	halt(8)
halt: stop the	processor	halt(8)
m4: macro	processor	m4(1)
reboot: reboot system or halt	processor	reboot(2)

prof: display profile data		prof(1)
5152 Graphics Printer. proff: nroff for the IBM 4201 Proprinter and IBM .		pp(1)
		proff(1)
		profil(2)
profil: execution time profile		profil(2)
monitor, monstartup, moncontrol: prepare execution profile		monitor(3)
kgmon: generate a dump of the operating system's profile buffers.		kgmon(8)
gprof: display call graph profile data		gprof(1)
prof: display profile data		prof(1)
pxp: Pascal execution profiler		pxp(1)
drtest: standalone disk test program		drtest(8)
finger: user information lookup program		finger(1)
liszt: compile a Franz Lisp program		liszt(1)
lpc: line printer control program		lpc(8)
lpq: spool queue examination program		lpq(1)
lxref: lisp cross reference program.		lxref(1)
msgs: system messages and junk mail program		msgs(1)
mt: magnetic tape manipulating program		mt(1)
mt: magnetic tape manipulating program.		mt(1)
pxref: Pascal cross-reference program		pxref(1)
rdist: remote file distribution program.		rdist(1)
syscall: system call interface program.		syscall(8)
timedc: timed control program.		timedc(8)
units: conversion program	• •	units(1)
end, etext, edata: last locations in program.	• •	end(3)
	• •	whereis(1)
	• •	xhost(1)
	• •	
	• •	xset(1)
cb: C program beautifier		cb(1)
only). which: locate a program file including aliases and paths (csh	• •	which(1)
make: maintain program groups.		make(1)
nice: set program priority.		nice(3C)
getpriority, setpriority: get/set program scheduling priority.		getpriority(2)
indent: indent and format C program source		indent(1)
ptfinstall: install a Program Temporary Fix (PTF)		ptfinstall(8)
assert: program verification		assert(3)
lint: a C program verifier.		lint(1)
pf: set keyboard program-function keys		pf(1)
fp: Functional Programming language compiler/interpreter	• •	fp(1)
lex: generator of lexical analysis programs.		lex(1)
struct: structure Fortran programs.		struct(1)
vgrind: grind nice listings of programs.		vgrind(1)
vgrind: grind nice listings of programs for the IBM 3812 Pageprinter		vgrind(1)
troff. vlp: Format Lisp programs to be printed with nroff, vtroff, or		vlp(1)
xstr: extract strings from C programs to implement shared strings		xstr(1)
vv: Proteon proNET 10 Megabit ring		vv(4)
xprop - X Window System property displayer		xprop(1)
colpro: column filter for IBM 4201 Proprinter		colpro(1)
ibmgra, ibmpro: output filters for the IBM 4201 Proprinter and IBM 5152 Graphics Printer. ibmbit,		lpfilter(8r)
		(**/ 1 \
		pron(1)
proff: nroff for the IBM 4201 Proprinter and IBM 5152 Graphics Printer	• •	proff(1) prff(1)
proff: nroff for the IBM 4201 Proprinter and IBM 5152 Graphics Printer post-processing filter. prfl: IBM 4201 Proprinter/IBM 5152 Graphics Printer nroff	• •	prfl(1)
proff: nroff for the IBM 4201 Proprinter and IBM 5152 Graphics Printer post-processing filter. prfl: IBM 4201 Proprinter/IBM 5152 Graphics Printer nroff vv: Proteon proNET 10 Megabit ring	• •	prfl(1) vv(4)
proff: nroff for the IBM 4201 Proprinter and IBM 5152 Graphics Printer post-processing filter. prfl: IBM 4201 Proprinter/IBM 5152 Graphics Printer nroff vv: Proteon proNET 10 Megabit ring		prfl(1) vv(4) arp(4P)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message Protocol		prff(1) vv(4) arp(4P) icmp(4P)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram Proprinter and IBM 5152 Graphics Printer. Proprinter IBM 5152 Graphics Printer. Proprinter and IBM 5152 Graphics Printer. Proprinter All IBM 5152 Graphics Printer and IBM 5152 Graphics Printer an		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram ip: Internet post-processing filter. prfl: IBM 4201 vv: Proprinter and IBM 5152 Graphics Printer. Proprinter and IBM 5152 Graphics Printer. Proprinter and IBM 5152 Graphics Printer. Protocol. Protocol. Protocol. Protocol. Proprinter and IBM 5152 Graphics Printer. Proprinter/IBM 5152 Graphics Printer. Protocol.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram ip: Internet rvd: Remote Virtual Disk post-processing filter. prfl: IBM 4201 vv: Proprinter and IBM 5152 Graphics Printer. Protocol.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram rvd: Remote Virtual Disk spp: Xerox Sequenced Packet post-processing filter. prfl: IBM 4201 vv: Proprinter and IBM 5152 Graphics Printer. Proprinter and IBM 5152 Graphics Printer. Proprinter and IBM 5152 Graphics Printer. Protocol.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control post-processing filter. prfl: IBM 4201 vv: Proprinter and IBM 5152 Graphics Printer. Proprinter/IBM 5152 Graphics Printer nroff Protocol. Protocol. Protocol. Protocol. Protocol. Protocol. Protocol. Protocol.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram ip: Internet rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram Protocol. Protocol. Proprinter and IBM 5152 Graphics Printer Protocol. Protocol. Protocol. Protocol. Protocol. Protocol. Protocol.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram ip: Internet rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram XNSrouted: NS Routing Information Proprinter and IBM 5152 Graphics Printer. Proprinter/IBM 5152 Graphics Printer. Protocol. Protocol. Protocol. Protocol. Protocol. Protocol. Protocol.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram ip: Internet rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram XNSrouted: NS Routing Information getprotobyname, setprotoent, endprotoent: get		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram protocol. rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram XNSrouted: NS Routing Information getprotobyname, setprotoent, endprotoent: get inet: Internet Proprinter and IBM 5152 Graphics Printer. Protocol.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram ip: Internet rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram XNSrouted: NS Routing Information getprotobyname, setprotoent, endprotoent: get inet: Internet ns: Xerox Network Systems(tm) Proprinter and IBM 5152 Graphics Printer. Protocol.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ns(4F)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram protocol. ip: Internet rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram XNSrouted: NS Routing Information getprotobyname, setprotoent, endprotoent: get inet: Internet ns: Xerox Network Systems(tm) ap: asynchronous data mode Protocol Immunication Protocol Immu		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ns(4F) ap(4)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram XNSrouted: NS Routing Information getprotobyname, setprotoent, endprotoent: get ns: Xerox Network Systems(tm) ap: asynchronous data mode rmt: remote magtape Proprinter and IBM 5152 Graphics Printer. Protocol. Pro		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ap(4) rmt(8C)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram XNSrouted: NS Routing Information getprotobyname, setprotoent, endprotoent: get inet: Internet ns: Xerox Network Systems(tm) ap: asynchronous data mode rmt: remote magtape protocol iname data base.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ns(4F) ap(4) rmt(8C) protocols(5)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram XNSrouted: NS Routing Information getprotobyname, setprotoent, endprotoent: get inet: Internet ns: Xerox Network Systems(tm) ap: asynchronous data mode rmt: remote magtape protocol server. Proprinter and IBM 5152 Graphics Printer. Proprinter/IBM 5152 Graphics Printer. Protocol. Proto		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ns(4F) ap(4) rmt(8C) protocols(5) ftpd(8C)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram ip: Internet rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram XNSrouted: NS Routing Information getprotobyname, setprotoent, endprotoent: get inet: Internet ns: Xerox Network Systems(tm) ap: asynchronous data mode rmt: remote magtape protocols: ftpd: DARPA Internet File Transfer telnetd: DARPA TELNET protocol server. Proprinter and IBM 5152 Graphics Printer. Proprinter, IBM 5152 Graphics Printer. Proprinter, IBM 5152 Graphics Printer. Proprinter, IBM 5152 Graphics Printer. Proprinter/IBM 5152 Graphics Printer. Protocol. Protocol aemon. Protocol family. Protocol		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ns(4F) ap(4) rmt(8C) protocols(5) ftpd(8C) telnetd(8C)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram ip: Internet rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram XNSrouted: NS Routing Information getprotobyname, setprotoent, endprotoent: get inet: Internet ns: Xerox Network Systems(tm) ap: asynchronous data mode rmt: remote magtape protocols ftpd: DARPA Internet File Transfer telnetd: DARPA TellNET tftpd: DARPA Trivial File Transfer rvo: IBM 5152 Graphics Printer. Protocol. Protocol daemon. Protocol family. Protocol server.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ns(4F) ap(4) rmt(8C) protocols(5) ftpd(8C) telnetd(8C) tftpd(8C)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram protocol. protocol aemon. protocol family. protocol family. protocol family. protocol family. protocol of aemon. protocol family. protocol of aemon. protocol family. protocol family. protocol family. protocol of aemon. protocol of ae		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) tcp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ns(4F) ap(4) rmt(8C) protocols(5) ftpd(8C) telnetd(8C) tftpd(8C) trpt(8C)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram ip: Internet rvd: Remote Virtual Disk spp: Xerox Sequenced Packet tcp: Internet Transmission Control udp: Internet User Datagram XNSrouted: NS Routing Information getprotobyname, setprotoent, endprotoent: get inet: Internet ns: Xerox Network Systems(tm) ap: asynchronous data mode rmt: remote magtape protocols: ftpd: DARPA Internet File Transfer telnetd: DARPA Trivial File Transfer trpt: transliterate trsp: transliterate sequenced packet reconstruction Protocol. Protocol ademon. Protocol family. Protocol line discipline. Protocol line discipline. Protocol server.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ap(4) rmt(8C) protocols(5) ftpd(8C) telnetd(8C) trpt(8C) trsp(8C)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ap(4) rmt(8C) protocols(5) ftpd(8C) telnetd(8C) trpt(8C) trsp(8C) protocols(5)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 post-processing filter. prfl: IBM 4201 arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram protocol. protocol daemon. protocol family. protocol family. protocol family. protocol family. protocol family. protocol module. protocol server. protocol race. protocols: protocol name data base.		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ap(4) rmt(8C) protocols(5) ftpd(8C) telnetd(8C) trpt(8C) trsp(8C) protocols(5) mkproto(8)
proff: nroff for the IBM 4201 post-processing filter. prfl: IBM 4201 vv: arp: Address Resolution icmp: Internet Control Message idp: Xerox Internet Datagram		prfl(1) vv(4) arp(4P) icmp(4P) idp(4P) ip(4P) rvd(4p) spp(4P) tcp(4P) udp(4P) XNSrouted(8C) getprotoent(3N) inet(4F) ap(4) rmt(8C) protocols(5) ftpd(8C) telnetd(8C) trpt(8C) trsp(8C) protocols(5)

false, true:	provide truth values	false(1)
true, false:	provide truth values	
device interface.	ps: Evans and Sutherland Picture System 2 graphics	
	ps: process status.	
pty:	pseudo terminal driver	
	psignal, sys_siglist: system signal messages	
	psp: planar serial port RS232C interface	psp(4)
	pstat: print system facts	• : :
ptfinstall: install a Program Temporary Fix	pstat: print system facts	pstat(8)
punistan, nistan a Program Temporary I'ix	ptfinstall: install a Program Temporary Fix (PTF)	ptfinstall(8) ptfinstall(8)
	ptrace: process trace	ptrace(2)
	ptrace: process trace	
	ptroff: print troff files on IBM 3812 Pageprinter	ptroff(1)
	ptx: permuted index	
	pty: pseudo terminal driver	
ungetc:	push character back into input stream	ungetc(3S)
pushd:	push shell directory stack	csh(1)
	pushd: push shell directory stack	csh(1)
puts, fputs:	put a string on a stream.	
putc, putchar, fputc, putw: unit.	put character or word on a stream.	
on a stream.	putc, fputc: write a character to a fortran logical putc, putchar, fputc, putw: put character or word	
stream. putc,	putchar, fputc, putw. put character of word putchar, fputc, putw: put character or word on a	i :
ottomin pato,	puts, fputs: put a string on a stream	
putc, putchar, fputc,	putw: put character or word on a stream	putc(3S)
1, 1,	pwd: working directory name	pwd(1)
	px: Pascal interpreter	px(1)
	pxp: Pascal execution profiler	pxp(1)
	pxref: Pascal cross-reference program	pxref(1)
qe: DEC DEQNA	Q-bus 10 Mb/s Ethernet interface	
	qe: DEC DEQNA Q-bus 10 Mb/s Ethernet interface	qe(4)
	qsort: quick sort.	qsort(3F)
VI_QFont, VI_QMerge, VI_QPoint, VI_QWidth:	query graphics parameters. /VI QColor, VI QDash,	qsort(3)
lprm: remove jobs from the line printer spooling	queue	query(3G) lprm(1)
insque, remque: insert/remove element from a	queue.	insque(3)
lpq: spool	queue examination program.	lpq(1)
atq: print the	queue of jobs waiting to be run	atq(1)
uucico, uucpd: transfer files	queued by uucp or uux	uucico(8C)
xemul: X input emulator for	queuing keyboard and mouse events	xemul(4)
qsort:	quick sort	qsort(3F)
qsort:	quicker sort.	qsort(3)
quotacheck: file system	quot: summarize file system ownership quota consistency checker	quot(8) quotacheck(8)
quotacheek. He system	quota display disc usage and limits.	quota(1)
	quota: manipulate disk quotas.	quota(2)
	quotacheck: file system quota consistency checker	quotacheck(8)
quotaon,	quotaoff: turn file system quotas on and off	quotaon(8)
off.	quotaon, quotaoff: turn file system quotas on and	quotaon(8)
edquota: edit user	quotas	edquota(8)
quota: manipulate disk	quotas.	quota(2)
repquota: summarize	quotas for a file system	repquota(8)
setquota: enable/disable	quotas on a file system.	setquota(2)
quotaon, quotaoff: turn file system	quotas on and off	quotaon(8) rand(3F)
	rand, srand: random number generator	rand(3C)
generator.	random, drandm, irandm: better random number	random(3F)
ranlib: convert archives to	random libraries.	ranlib(1)
rand, srand:	random number generator	rand(3C)
random, drandm, irandm: better	random number generator	random(3F)
random, srandom, initstate, setstate: better	random number generator; routines for changing/	random(3)
number generator; routines for changing/	random, srandom, initstate, setstate: better random	random(3)
rand, drand, irand: return	random values.	rand(3F)
	ranlib: convert archives to random libraries	ranlib(1)
ratfor:	ratfor: rational Fortran dialect	ratfor(1) ratfor(1)
imp: IMP	raw socket interface.	imp(4P)
mp. mi	re: command script for auto-reboot and daemons.	rc(8)
	rc.config: configuration file for startup scripts	rc.config(5)
stream to a remote command.	rcmd, rresvport, ruserok: routines for returning a	rcmd(3)
	rdist: remote file distribution program	rdist(1)
, m	rdump: file system dump across the network	rdump(8C)
tbuffer: streaming tape buffered	read	tbuffer(8)
getpass:	read a password	getpass(3)
source:	read commands from file	csh(1)

WI MOI WI EDI	and display too	1/2/2)
VI_MRead, VI_FRead: read, readv:	read display data	read(3G)
/continue, cd, eval, exec, exit, export, login,	read input	read(2) sh(1)
pontinuo, ed, evan, exce, expert, rogin,	read, readv: read input.	read(2)
readlink:	read value of a symbolic link	readlink(2)
dosread:	read, write, dir, delete on PC-DOS diskette	dosread(1)
directory operations. opendir,	readdir, telldir, seekdir, rewinddir, closedir:	directory(3)
open: open a file for	reading or writing, or create a new file	open(2)
command/ /od aval avac svit avacet login road	readlink: read value of a symbolic link	readlink(2)
command/ /cd, eval, exec, exit, export, login, read, read,	readonly, set, shift, times, trap, umask, wait: readv: read input	sh(1) read(2)
bad144:	read/write dec standard 144 bad sector information.	bad144(8)
lseek: move	read/write pointer	lseek(2)
setregid: set	real and effective group ID	setregid(2)
setreuid: set	real and effective user ID's	setreuid(2)
wm: a simple	real-estate-driven window manager	wm(1)
malloc, free,	realloc, calloc, alloca: memory allocator.	malloc(3)
symorder:	rearrange name list	symorder(1) reboot(8)
	reboot: reboot system or halt processor.	reboot(2)
reboot:	reboot system or halt processor	reboot(2)
	reboot: UNIX bootstrapping procedures	reboot(8)
fastboot, fasthalt:	reboot/halt the system without checking the disks	fastboot(8)
newaliases:	rebuild the data base for the mail aliases file	newaliases(1)
recv, recvfrom, recvmsg:	receive a message from a socket	recv(2)
mail: send and binmail: send or	receive mail.	mail(1)
rmail: handle remote mail	receive mail among users	binmail(1) rmail(1)
Than hande femote man	re comp, re exec: regular expression handler	regex(3)
rehash:	recompute command hash table	csh(1)
fdisk: boot	record partition table maintenance utility	fdisk(8)
utmp, wtmp: login	records	utmp(5)
VI_Tile: tile a socket.	rectangle.	tile(3G)
recv,	recv, recvfrom, recvmsg: receive a message from a recvfrom, recvmsg: receive a message from a socket	recv(2) recv(2)
recv, recvfrom,	recvmsg: receive a message from a socket	recv(2)
eval:	re-evaluate shell data	csh(1)
re_comp,	re_exec: regular expression handler	regex(3)
documents.	refer: find and insert literature references in	refer(1)
lxref: lisp cross build inverted index for a bibliography, find	reference program	lxref(1) lookbib(1)
refer: find and insert literature	references in documents.	refer(1)
xrefresh -	refresh all windows on the screen	xrefresh(1)
re_comp, re_exec:	regular expression handler	regex(3)
•	rehash: recompute command hash table	csh(1)
comm: select or	reject lines common to two sorted files	comm(1)
lorder: find ordering join:	relation for an object library	lorder(1) join(1)
sigpause: atomically	release blocked signals and wait for interrupt.	sigpause(2)
strip: remove symbols and	relocation bits.	strip(1)
copysign, drem, finite, logb, scalb: copysign,	remainder, exponent manipulations.	ieee(3M)
leave:	remind you when you have to leave	leave(1)
calendar:	reminder service	calendar(1)
ruserok: routines for returning a stream to a rexec: return stream to a	remote command. rcmd, rresvport,	rcmd(3) rexec(3)
L.cmds: UUCP	remote command permissions file.	L.cmds(5)
rexecd:	remote execution server	rexecd(8C)
rdist:	remote file distribution program	rdist(1)
L.sys: UUCP	remote host description file.	L.sys(5)
remote:	remote host description file.	remote(5)
phones: rlogind:	remote host phone number data base remote login server	phones(5) rlogind(8C)
tn3270: full-screen	remote login to IBM VM/CMS	tn3270(1)
tn3270: full-screen	remote login to IBM VM/CMS	tn3270(1)
rmt:	remote magtape protocol module	rmt(8C)
rmail: handle	remote mail received via uucp	rmail(1)
rshd:	remote: remote host description file	remote(5) rshd(8C)
talkd:	remote user communication server.	talkd(8C)
fingerd:	remote user information server.	fingerd(8C)
uupoll: poll a	remote UUCP site	uupoll(8C)
rvd:	Remote Virtual Disk protocol.	rvd(4p)
newvd: create a new filesystem on a	Remote Virtual Disk (RVD)	newvd(8)
vdspin, vdspind: spin up or spin down a vddb:	Remote Virtual Disk (RVD)	vdspin(2) vddb(8)
	Remote Virtual Disk (RVD) pack	rvddown(8)
· · · · · · · · · · · · · · · · · · ·	, , , , , , , , , , , , , , , , , , ,	` '

	m	. (0)
spinup, spindown: spin up/down	Remote Virtual Disk (RVD) pack	spinup(8)
rvdexch: exchange names of two	Remote Virtual Disk (RVD) packs	rvdexch(8)
rvdflush: spindown client's	Remote Virtual Disk (RVD) packs	rvdflush(8)
savervd, zaprvd, savephys: back up and restore	Remote Virtual Disk (RVD) packs to and from tape	savervd(8)
rvdchlog: change logging level of	Remote Virtual Disk (RVD) server	rvdchlog(8)
rvdgetm: get operations message from	Remote Virtual Disk (RVD) server	rvdgetm(8)
rvdsend - send control stream to	Remote Virtual Disk (RVD) server	rvdsend(8)
rvdsetm: set operations message on	Remote Virtual Disk (RVD) server	rvdsetm(8)
rvdshow: show connections to	Remote Virtual Disk (RVD) server	rvdshow(8)
rvdshut: force shutdown of	Remote Virtual Disk (RVD) server	rvdshut(8)
table. rvddb:	Remote Virtual Disk (RVD) server configuration	rvddb(5)
rvdsrv:	Remote Virtual Disk (RVD) server daemon.	rvdsrv(8)
rvdlog: cause	Remote Virtual Disk (RVD) server to log statistics	rvdlog(8)
vdstats: acquire client	Remote Virtual Disk (RVD) statistics	vdstats(2)
vdstats: list client	Remote Virtual Disk (RVD) statistics	vdstats(8)
up, down: client	Remote Virtual Disk (RVD) utilities	up(1)
/etc/rvd/rvdtab: information about client	Remote Virtual Disks (RVDs)	rvdtab(5)
unlink:	remove a directory entry.	unlink(3F)
rmdir:	remove a directory file	rmdir(2)
unalias:	remove aliases.	csh(1)
flock: apply or	remove an advisory lock on an open file	flock(2)
colrm:	remove columns from a file	colrm(1)
unlink:	remove directory entry.	unlink(2)
unsetenv:	remove environment variables	csh(1)
mount, umount: mount or	remove file system.	mount(2)
unifdef:	remove ifdef'ed lines.	unifdef(1)
lprm:	remove jobs from the line printer spooling queue	lprm(1)
atrm:	remove jobs spooled by at.	atrm(1)
deroff:	remove nroff, troff, tbl and eqn constructs	deroff(1)
unlimit:	remove resource limitiations.	csh(1)
strip:	remove symbols and relocation bits	strip(1)
rmdir, rm:	remove (unlink) directories or files	rmdir(1)
rm, rmdir:	remove (unlink) files or directories	rm(1)
insque,	remque: insert/remove element from a queue	insque(3)
rename:	rename a file.	rename(3F)
	rename: change the name of a file	rename(2)
mv: move or	rename files.	mv(1)
	rename: rename a file.	rename(3F)
	renice: alter priority of running processes	reniœ(8)
fsck: file system consistency check and interactive	repair	fsck(8)
trpfpe, fpecnt: trap and	repair floating point faults	trpfpe(3F)
trapov: trap and	repair floating point overflow	trapov(3F)
while:	repeat commands conditionally	csh(1)
	repeat: execute command repeatedly	csh(1)
uniq: report	repeated lines in a file	uniq(1)
repeat: execute command	repeatedly.	csh(1)
yes: be	repetitively affirmative	yes(1)
iostat:	report I/O statistics	iostat(1)
uniq:	report repeated lines in a file	uniq(1)
sendbug: mail a system bug	report to 4bsd-bugs	sendbug(1)
vmstat:	report virtual memory statistics	vmstat(1)
vmstat:	report virtual memory statistics	vmstat(1)
bugfiler: file bug	reports in folders automatically.	bugfiler(8)
fseek, ftell:	reposition a file on a logical unit.	fseek(3F)
fseek, ftell, rewind:	reposition a stream	fseek(3S)
Today, Today, Tollian	repquota: summarize quotas for a file system.	repquota(8)
notify:	request immediate notification.	csh(1)
lock:	reserve a terminal	lock(1)
disk: format of	reserved areas of the hard disk.	disk(4)
res mkquery, res send,	res_init, dn_comp, dn_expand: resolver routines	resolver(3)
scale:	resize a bitmap image	scale(1)
dn expand: resolver routines.	res_mkquery, res_send, res_init, dn_comp,	resolver(3)
arp: address	resolution display and control	arp(8C)
arp: Address	Resolution Protocol.	arp(4P)
arp. riddress	resolver configuration file.	resolver(5)
res send, res init, dn comp, dn expand:	resolver routines. res mkquery,	resolver(3)
getrlimit, setrlimit: control maximum system	resource consumption	getrlimit(2)
vlimit: control maximum system	resource consumption.	vlimit(3C)
xrdb - Server	Resource Database Utility	xrdb(1)
limit: alter per-process	resource limitations.	csh(1)
unlimit: remove	resource limitiations.	csh(1)
getrusage: get information about	resource utilization.	getrusage(2)
vtimes: get information about	resource utilization.	vtimes(3C)
routines. res mkquery,	res send, res init, dn comp, dn expand: resolver	resolver(3)
restore: incremental file system	restore	restore(8)
restore: incremental file system	restore	restore(8)
receive merelinent me bystem		

rrestore:	restore a nie system dump across the network	rrestore(8C)
	restore: incremental file system restore	
	restore: incremental file system restore	` '
tape. savervd, zaprvd, savephys: back up and	restore Remote Virtual Disk (RVD) packs to and from	savervd(8)
network. restore.tape,	restore.net: install system from tape or over	restore.tape(8)
or over network.	restore.tape, restore.net: install system from tape	
suspend: suspend a shell,	resuming its superior	
mset:	retrieve ASCII to IBM 3270 keyboard map	mset(1)
mset:	retrieve ASCII to IBM 3270 keyboard map	2.1
getfpemulator:	return address of the floating-point emulator	
getarg, iargo:	return command line arguments	
fdate:	return date and time in an ASCII string	fdate(3F)
idate, itime:	return date or time in numerical form	idate(3F)
etime, dtime:	return elapsed execution time	etime(3F)
flmin, flmax, ffrac, dflmin, dflmax, dffrac, inmax:	return extreme values	
sigreturn:	return from signal	
•		
vacation:	return "I am on vacation" indication.	
getfloatstate:	return machine and process floating point state	
rand, drand, irand:	return random values	rand(3F)
rexec:	return stream to a remote command	rexec(3)
time, ctime, ltime, gmtime:	return system time	time(3F)
loc:	return the address of an object	
rcmd, rresvport, ruserok: routines for		
rema, ries vport, ruserok. routilies for		
1.01.	rev: reverse lines of a file	` '
col: filter	reverse line feeds	col(1)
rev:	reverse lines of a file	rev(1)
lastcomm: show last commands executed in	reverse order	lastcomm(1)
fseek, ftell,	rewind: reposition a stream	
opendir, readdir, telldir, seekdir,	rewinddir, closedir: directory operations	
opendir, readdir, tendir, seekdir,	· · · · · · · · · · · · · · · · · · ·	• • •
	rexec: return stream to a remote command	• • • • • • • • • • • • • • • • • • • •
	rexecd: remote execution server	` '
index,	rindex, Inblnk, Ien: tell about character objects	
strcmp, strncmp, strcpy, strncpy, strlen, index,	rindex: string operations. strcat, strncat,	string(3)
vv: Proteon proNET 10 Megabit	ring	vv(4)
finite, infinity,/ copysign, drem, logb, scalb,	rint, classdouble, classfloat, isnan, unordered,	. `
lptest: generate lineprinter	ripple pattern.	
hk: RK6-11/RK06 and	RK07 moving head disk	
hk:	RK6-11/RK06 and RK07 moving head disk	· ,
crl: VAX 8600 console	RL02 interface	
	rlogind: remote login server	rlogind(8C)
rmdir,	rm: remove (unlink) directories or files	rmdir(1)
,	rm, rmdir: remove (unlink) files or directories	rm(1)
	rmail: handle remote mail received via uucp	
	rmdir: remove a directory file.	
1111,	rmdir: remove (unlink) files or directories	``
	rmdir, rm: remove (unlink) directories or files	
	rmt: remote magtape protocol module	
	roffbib: run off bibliographic database	roffbib(1)
cbrt, sqrt: cube root, square	root	sqrt(3M)
chroot; change	root directory.	
cbrt, sqrt: cube	root, square root.	sqrt(3M)
xsetroot: X window system	root window parameter setting utility	
mem, kmem, kmem1, kmem2, kmem4,	ros, aspamem: main memory.	
	route: manually manipulate the routing tables	
	routed: network routing daemon	routed(8C)
inet netof: Internet address manipulation	routines. /inet_ntoa, inet_makeaddr, inet_lnaof,	inet(3N)
ns addr, ns ntoa: Xerox NS(tm) address conversion	routines	
res_send, res_init, dn_comp, dn_expand: resolver	routines. res mkquery,	
tgoto, tputs: terminal independent operation		
	routines. tgetent, tgetnum, tgetflag, tgetstr,	
setstate: better random number generator;	routines for changing generators. /initstate,	
command. rcmd, rresvport, ruserok:	routines for returning a stream to a remote	
Xtext:	routines to provide simple text output windows	
Xtty:	routines to provide terminal emulator windows	
routed: network	routing daemon	routed(8C)
XNSrouted: NS	Routing Information Protocol daemon	XNSrouted(8C)
route: manually manipulate the	routing tables.	
mout,/ madd, msub, mult, mdiv, pow, gcd, invert,	rpow, msqrt, mcmp, move, min, omin, fmin, m_in,	
network.	rrestore: restore a file system dump across the	
to a remote command. rcmd,	rresvport, ruserok: routines for returning a stream	rcmd(3)
asy: multi-port asynchronous communications	RS232C interface	asy(4)
psp: planar serial port	RS232C interface	psp(4)
	rshd: remote shell server	
bit: and, or, xor, not,		
debug: debugger for the IBM	RT PC.	
kbdlock: lock the keyboard of the IBM	RT PC.	
un: IBM	RT PC Baseband Adapter for use with Ethernet	un(4)

lan: IBM	RT PC Token-Ring Adapter	lan(4)
atq: print the queue of jobs waiting to be	run	atq(1)
nice, nohup:	run a command at low priority (sh only)	niœ(1)
nohup:	run command immune to hangups	csh(1)
nice:		csh(1)
roffbib:	run off bibliographic database	roffbib(1)
gcore: get core images of	running processes	gcore(1)
renice: alter priority of	running processes	renice(8)
remote command. rcmd, rresvport,	ruserok: routines for returning a stream to a	rcmd(3)
create a new filesystem on a Remote Virtual Disk	(RVD). newvd:	newvd(8)
vdspind: spin up or spin down a Remote Virtual Disk	(RVD). vdspin,	vdspin(2)
vddb: Remote Virtual Disk	(RVD) data base manager	vddb(8)
rvdcopy: copy contents of one	RVD disk pack to another	rvdcopy(8)
rvddown: force spindown of a Remote Virtual Disk	(RVD) pack	rvddown(8)
spinup, spindown: spin up/down Remote Virtual Disk	(RVD) pack	spinup(8)
rvdexch: exchange names of two Remote Virtual Disk	(RVD) packs	rvdexch(8)
rvdflush: spindown client's Remote Virtual Disk	(RVD) packs	rvdflush(8)
savephys: back up and restore Remote Virtual Disk	(RVD) packs to and from tape. savervd, zaprvd,	savervd(8)
	rvd: Remote Virtual Disk protocol.	rvd(4p)
change logging level of Remote Virtual Disk	(RVD) server. rvdchlog:	rvdchlog(8)
get operations message from Remote Virtual Disk	(RVD) server. rvdgetm:	rvdgetm(8)
- send control stream to Remote Virtual Disk	(RVD) server. rvdsend	rvdsend(8)
set operations message on Remote Virtual Disk	(RVD) server. rvdsetm:	rvdsetm(8)
rvdshow: show connections to Remote Virtual Disk	(RVD) server	rvdshow(8)
rvdshut: force shutdown of Remote Virtual Disk	(RVD) server	rvdshut(8)
rvddb: Remote Virtual Disk	(RVD) server configuration table	rvddb(5)
rvdsrv: Remote Virtual Disk	(RVD) server daemon	rvdsrv(8)
rvdlog: cause Remote Virtual Disk	(RVD) server to log statistics	rvdlog(8)
rvdhosts: print a list of	RVD servers	rvdhosts(8)
vdstats: acquire client Remote Virtual Disk	(RVI) statistics	vdstats(2)
vdstats: list client Remote Virtual Disk	(RVD) statistics	vdstats(8)
up, down: client Remote Virtual Disk	(RVD) utilities	up(1)
Disk (RVD) server.	rvdchlog: change logging level of Remote Virtual	rvdchlog(8)
another.	rvdcopy: copy contents of one RVD disk pack to	rvdcopy(8)
configuration table.	rvddb: Remote Virtual Disk (RVD) server	rvddb(5)
(RVD) pack.	rvddown: force spindown of a Remote Virtual Disk	rvddown(8)
(RVD) packs.	rvdexch: exchange names of two Remote Virtual Disk .	rvdexch(8)
(RVD) packs.	rvdflush: spindown client's Remote Virtual Disk	rvdflush(8)
Disk (RVD) server.	rvdgetm: get operations message from Remote Virtual .	rvdgetm(8)
Diox (14 1 D) 501 701.	rvdhosts: print a list of RVD servers	rvdhosts(8)
log statistics.	rvdlog: cause Remote Virtual Disk (RVD) server to	rvdlog(8)
information about client Remote Virtual Disks	(RVDs). /etc/rvd/rvdtab:	rvdtab(5)
Disk (RVD) server.	rvdsend - send control stream to Remote Virtual	rvdsend(8)
Disk (RVD) server.	rvdsetm: set operations message on Remote Virtual	rvdsetm(8)
(RVD) server.	rvdshow: show connections to Remote Virtual Disk	rvdshow(8)
(RVD) server.	rvdshut: force shutdown of Remote Virtual Disk	rvdshut(8)
(1715) 501 701.	rvdsrv: Remote Virtual Disk (RVD) server daemon	rvdsrv(8)
	rwhod: system status server	rwhod(8C)
	rx: DEC RX02 floppy disk interface	
rx: DEC		
12. 000	rxformat: format floppy disks	rxformat(8V)
	sa, accton: system accounting.	
	sautil: standalone utility package.	sautil(8r)
savecore:	save a core dump of the operating system	
007000101	savecore: save a core dump of the operating system	savecore(8)
(RVD) packs to and from tape. savervd, zaprvd,	savephys: back up and restore Remote Virtual Disk	savervd(8)
Remote Virtual Disk (RVD) packs to and from tape.	savervd, zaprvd, savephys: back up and restore	savervd(8)
brk,	sbrk: change data segment size.	brk(2)
System Interface (SCSI) Adapter.	sc: IBM 9332 disks using the IBM Small Computer	sc(4)
copysign, drem, finite, logb,	scalb: copysign, remainder, exponent manipulations	ieee(3M)
unordered, finite, infinity,/ copysign, drem, logb,	scalb, rint, classdouble, classfloat, isnan,	ieee(3)
unordered, mine, minity,/ copysign, drein, logo,	scale: resize a bitmap image.	scale(1)
scandir, alphasort:	scan a directory.	scale(1) scandir(3)
		keyboard codes(5)
keyboard_codes: keyboard	scancode table	scandir(3)
a	scanf, fscanf, sscanf: formatted input conversion	scanf(3S)
awk: pattern	scanning and processing language	awk(1)
	sccs: front end for the SCCS subsystem	sccs(1)
sccs: front end for the	SCCS subsystem.	sccs(1)
alarm:	schedule signal after specified time.	alarm(3C)
ualarm:	schedule signal after specified time.	ualarm(3)
getpriority, setpriority: get/set program	scheduling priority.	getpriority(2)
xcalc: X based	scientific calculator.	xcalc(1)
clear: clear terminal	screen	clear(1)
xrefresh - refresh all windows on the	screen	xrefresh(1)
setscreen: control display	screen access.	setscreen(8)

VI_Color: change		color(3G)
curses: ex. vi:	screen functions with "optimal" cursor motion	curses(3X)
consoles: utility database of display	screen oriented (visual) display editor based on screens	vi(1) consoles(5)
rc: command	script for auto-reboot and daemons.	rc(8)
	script: make typescript of terminal session	script(1)
onintr: process interrupts in command	scripts	csh(1)
rc.config: configuration file for startup	scripts	rc.config(5)
disks using the IBM Small Computer System Interface	(SCSI) Adapter. sc: IBM 9332	sc(4)
turing and forth on the course and forth of	scsiformat: format the IBM 9332 disk unit.	scsiformat(8c)
/min, omin, fmin, m_in, mout, omout, fmout, m_out, grep, egrep, fgrep:	sdiv, itom: multiple precision integer arithmetic search a file for a pattern	mp(3X) grep(1)
xsend, xget, enroll:	secret mail.	xsend(1)
bad144: read/write dec standard 144 bad	sector information.	bad144(8)
badsect: create files to contain bad	sectors	badsect(8)
badsect: create files to contain bad	sectors	badsect(8)
	sed: stream editor.	sed(1)
opendir, readdir, telldir,	seekdir, rewinddir, closedir: directory operations	directory(3)
brk, sbrk: change data VI_Font, VI_GetFont, VI_DropFont:	segment size	brk(2)
vi_ront, vi_Getront, vi_Dropront.	select or reject lines common to two sorted files	font(3G) comm(1)
Commi	select: synchronous I/O multiplexing	select(2)
case:	selector in switch.	csh(1)
Academic Information Systems experimental display	self-tests. aedtest: IBM	aedtest(8)
send, sendto, sendmsg:	send a message from a socket	send(2)
kill:	send a signal to a process.	kill(3F)
mail:	send and receive mail	mail(1)
sendapar: server, rydsend -	send APAR	sendapar(8) rvdsend(8)
ping:	send ICMP ECHO REQUEST packets to network hosts.	ping(8)
sendmail:	send mail over the internet	sendmail(8)
binmail:	send or receive mail among users.	binmail(1)
socket. kill:	send, sendto, sendmsg: send a message from a send signal to a process	send(2) kill(2)
killpg:	send signal to a process group.	killpg(2)
1.9.	sendapar: send APAR	sendapar(8)
	sendbug: mail a system bug report to 4bsd-bugs	sendbug(1)
aliases: aliases file for	sendmail.	aliases(5)
sand sandto	sendmail: send mail over the internet	sendmail(8) send(2)
send, sendto, send,	sendmsg: send a message from a socket sendto, sendmsg: send a message from a socket	send(2)
diction, explain: print wordy	sentences; thesaurus for diction.	diction(1)
spp: Xerox	Sequenced Packet Protocol	spp(4P)
trsp: transliterate	sequenced packet protocol trace	trsp(8C)
slattach: attach psp: planar	serial lines as network interfaces serial port RS232C interface	slattach(8C) psp(4)
comsat: biff	server.	comsat(8C)
fingerd: remote user information	server.	fingerd(8C)
ftpd: DARPA Internet File Transfer Protocol	server	ftpd(8C)
ibm3812pp: IBM 3812 Pageprinter	server	ibm3812pp(8)
named: Internet domain name	server	named(8)
rexecd: remote execution rlogind: remote login	server	rexecd(8C) rlogind(8C)
rshd: remote shell	server.	rshd(8C)
change logging level of Remote Virtual Disk (RVD)	server. rvdchlog:	rvdchlog(8)
operations message from Remote Virtual Disk (RVD)	server. rvdgetm: get	rvdgetm(8)
- send control stream to Remote Virtual Disk (RVD)	server. rvdsend	rvdsend(8)
set operations message on Remote Virtual Disk (RVD) show connections to Remote Virtual Disk (RVD)	server. rvdshow:	rvdsetm(8) rvdshow(8)
force shutdown of Remote Virtual Disk (RVD)	server. rvdshut:	rvdshut(8)
rwhod: system status	server.	rwhod(8C)
talkd: remote user communication	server	talkd(8C)
telnetd: DARPA TELNET protocol		telnetd(8C)
	server	
tftpd: DARPA Trivial File Transfer Protocol	server	tftpd(8C)
rvddb: Remote Virtual Disk (RVD)	server	tftpd(8C) rvddb(5)
	server	tftpd(8C)
rvddb: Remote Virtual Disk (RVD) rvdsrv: Remote Virtual Disk (RVD) timed: time xrdb -	server	tftpd(8C) rvddb(5) rvdsrv(8) timed(8) xrdb(1)
rvddb: Remote Virtual Disk (RVD) rvdsrv: Remote Virtual Disk (RVD) timed: time xrdb - rvdlog: cause Remote Virtual Disk (RVD)	server	tftpd(8C) rvddb(5) rvdsrv(8) timed(8) xrdb(1) rvdlog(8)
rvddb: Remote Virtual Disk (RVD) rvdsrv: Remote Virtual Disk (RVD) timed: time xrdb -	server. server configuration table. server daemon. server daemon. Server Resource Database Utility server to log statistics. servers.	tftpd(8C) rvddb(5) rvdsrv(8) timed(8) xrdb(1) rvdlog(8) rvdhosts(8)
rvddb: Remote Virtual Disk (RVD) rvdsrv: Remote Virtual Disk (RVD) timed: time xrdb - rvdlog: cause Remote Virtual Disk (RVD) rvdhosts: print a list of RVD	server. server configuration table. server daemon. server daemon. Server Resource Database Utility server to log statistics. servers. services: service name data base.	tftpd(8C) rvddb(5) rvdsrv(8) timed(8) xrdb(1) rvdlog(8) rvdhosts(8) services(5)
rvddb: Remote Virtual Disk (RVD) rvdsrv: Remote Virtual Disk (RVD) timed: time xrdb - rvdlog: cause Remote Virtual Disk (RVD)	server. server configuration table. server daemon. server daemon. Server Resource Database Utility server to log statistics. servers.	tftpd(8C) rvddb(5) rvdsrv(8) timed(8) xrdb(1) rvdlog(8) rvdhosts(8)
rvddb: Remote Virtual Disk (RVD) rvdsrv: Remote Virtual Disk (RVD) timed: time xrdb rvdlog: cause Remote Virtual Disk (RVD) rvdhosts: print a list of RVD logout: end script: make typescript of terminal stty, gtty:	server. server configuration table. server daemon. server daemon. Server Resource Database Utility server to log statistics. services: service name data base. session. session. set and get terminal state (defunct).	tftpd(8C) rvddb(5) rvdsrv(8) timed(8) xrdb(1) rvdlog(8) rvdhosts(8) services(5) csh(1) script(1) stty(3C)
rvddb: Remote Virtual Disk (RVD) rvdsrv: Remote Virtual Disk (RVD) timed: time xrdb rvdlog: cause Remote Virtual Disk (RVD) rvdhosts: print a list of RVD logout: end script: make typescript of terminal	server. server configuration table. server daemon. server daemon. Server Resource Database Utility server to log statistics. servers. services: service name data base. session. session.	tftpd(8C) rvddb(5) rvdsrv(8) timed(8) xrdb(1) rvdlog(8) rvdhosts(8) services(5) csh(1) script(1)

VI_Clip:			
signetimaki: ummax: set current signal mask. ummax: signetimaki: ummax: set file creation mode mask. ummax: set file results mode mask. ummax: set file results mode mask. ummax: set file times. utiline(3C) ummaxi(2)	VI Clip:	set clipping window.	clip(3G)
utimes: utimes: utimes: set file times. utimes: set file times. set grows: refle times. set grows: set line dash pattern. set line dash pa			
selgroups: selgroups selgroups and proposes list. selgroups: selgroups apply apply a command to a server, redsetm: selforch,			• • •
setgroups: y 1 Dabl. y 1 Dabl. y 1 Dabl. y 1 Dabl. y 2 Dabl. y 2 Dabl. y 3 Dabl. y 3 Dabl. y 4 Dabl. y 4 Dabl. y 5 Dabl. y 6 Dabl. y 7 Dabl. y 7 Dabl. y 8 Dabl. y 1 Dabl. y 2 Dabl. y 1 Dabl. y 3 Dabl. y 4 Dabl. y 6 Dabl. y 6 Dabl. y 7 Dabl. y 1	utime:	set file times.	utime(3Ć)
y U Dabit set line width. y U Dabit set line width. apply: apply a command to a server. redetin: getsockopt, setsockopt: get and hostid. apply: apply a command to a server. redetin: getsockopt, setsockopt: getsockopt: getsockopt. getsockopt, setsockopt: getsockopt: getsockopt. hostid.			
V1 Dah: v1 Width: v1 Width: set line width hattern		set group access list.	
VI Width: apply: apply a command to a server, rotein gets eath enge and engegacy apply (1) and account of the server, rotein gets of arguments. server, rotein gets of arguments. set of print identifier of current host system. boottame(1) set process group. set build, set set date. date(1)	•.		
were personal poly: apply a command to a server. rodsettric server. Research se			
apply; apply a command to a server, rotest set of arguments: server rotest set of arguments: server rotest set operations message on Remote Virtual Disk (RVD) settering setting settering setting set	· · · · · · · · · · · · · · · · · · ·		
server, rvdsettmis getsockopt;			
getbockopt, setsockopt; get and hostname: set by: set or print name of current host system. hostname: set print name of current host system. hostname! set print name of current host system. set the date. set the date. date: print name of set the date. set the date. set the date. date: print name of set the date. set the date. set the date. date: print name of set the date. set wer and group ID. settled, set user and group ID. settled, set get get get get get get get get get g			* * * * /
hostide: hostname; hostnam			
set or print name of current host system. settigr present present process group. setting devel, exec, exit, export, login, read, readory, set, set, set, and effective group ID. set read and effective group ID. set very and group ID. set very experimental tabs. set the date. date; print and date; print and set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date. set under and group ID. set under the date.			
setpgrp noises group. setpgrp noises group. setpgrp noises group. set process group. set process group. set process group. set at an and effective user I'Ds. set process group. set and effective set prob. set process group. set and additive print and set print and	_		
setregid(2) setreuids: eval, exec, exit, export, login, read, readonly, setreuids: eval, exec, exit, export, login, read, readonly, styr styr styr styr styr styr styr styr	setpgrp:		
setreuid; eval, exec, exit, export, login, read, readouly, settinial options. stream of the date: print and date: print and date: print and setuid, setuid, setgid, setgid, setgid, setgid, settinial options. stream of the date: mint and set set in the date: mint and set in the	nice:		
eval, exec, exit, export, login, read, readonly, sty: sty: sty: sty: tabs: date: print and date: print and date: print and setuid, setuid, setruid, setgid, setgid; setgid: seture. stream. s	setregid:		setregid(2)
getty: sty: sty: sty: sty: sty: sty: sty:			
setuid, seteuid, setruid, setgid, setgid; set user and group ID. setuid, setgid, setgid, setgid, setgid; set user and group ID. setuid, setgid, setgid, setgid, setgid, setgid; set user and group ID. setuid, setgid, setgid, setgid, setgid; set user and group ID. setuid, setgid, setgid, setgid; set get group fice entry. getgroups: set group secss list. setgroups: setg			• •
setuid, setuid, setruid, setgid, setgid; seterninat and setuid, setuid, setuid, setuid, setgid, setgid; seterninat setuid; setuid, setuid, setuid, setuid, setgid, setgid; setgid: seturan. stream. setuid; setuid, setuid, setgid, setgid; setgid: set user and group ID. setuid; setuid; setuid, setuid, setgid, setgid; setgid: set user and group ID. setuid; setuid; setuid, set	•		- · · ·
date: print and date: print and setuid, setuid, setgid, setgid, setgid; seteny: a stream. s	•		
setuid, setuid, setuid, setuid, setejd, setegid. setuid, setui			
setuid, seteuid, setrejid, setegid, setegid, setegid, setegid, setepid setuid,			
setenty. a stream. stream. stream. steron. setuid, setuid, setuid, setgid, se			
setuid, seteuid, setruid, setgid, setsiden, setbuffer, setlinebuft assign buffering to a setbuff(s)		set user and group 1D	setuiu(3)
setuid, setuid		set variable in environment	cethuff3S)
setuid, seteuid, setruid, setgid, setregid, setren, set variable in environment. csh(1) user and group ID. setuid, entry. getfsent, getfsfile, getfsfile, getfstype, setuid, seteuid, setruid, setgid, setgid; set user and group ID. setuid(3) setgern, getgreid, getgram, setgen, getgreid, getgram, setgen, setgid, setgid; set user and group ID. setuid(3) gethostbyname, gethostbydaddr, gethostent, host. getbostid, getspens, setgroup secses islat. setgroups(3) gethostbyname, gethostbyname, gettimer, getget unique identifier of current gethostid(2) gettimer, getget value of interval timer. gettimer getgets value of interval timer. gettimer, syslog, openiog, closelog, getnetent, getnetbyaddr, getnetbyname, getgerpotoent, getprotobynumber, getpriority, getgerpotoent, getprotobynumber, getpriority, getprotoent, getprotobynumber, getpriority, getprotoent, getprotobynumber, getpriority, setport, getspens, getpriority, getprotoent, getprotopynumber, getpriority, setprocess group. getprotoent, getprotobynumber, getpriority, getprotoent, getprotopynumber, getpriority, setprocess group. getprotoent, getprotobynumber, getpriority, getprotoent, endprotoent; get protocol entry. getprotoent, getprotopynumber, getpriority, setprocess group. getprotoent, getprotopynumber, getprotopynu			
user and group ID. setuid, entry. getfsent, getfstepe, getfstype, setuid, setuid gethostmam, gethostmam, gethostmam, gethostmer, getitimer, getsent unique identifier of current sethostid, gethostmam, gethostmer, getitimer, getsent value of interval timer. gethostmer(2) sethostmer, getyper, sethostmer, sethostmer, getyper, sethostmer, sethostmer, getyper, sethostmer, getyper, sethostmer, getyper, sethostmer, sethostmer, getyper, sethostmer, sethostmer, getyper, sethostmer, getyper, sethostmer, getyper, sethostmer, sethostmer, sethostmer, sethostmer, sethostmer, sethostmer, getyper, sethostmer, sethostmer, getyper, sethostmer, sethostmer, sethostmer, sethostmer, sethostmer, sethostmer, sethostmer, sethostmer, getyper, sethostmer, s			
user and group ID. setuid, setroid, setroid set user and group ID. setuid(3) setgrent, getgrent, getgrent, getgrent, getgrent get group file entry. getproint) setgroups: set group access list. setgroups: setproint gethostiby activation of interval timer. gethostidy. Sethostic get; sethostid, get; sethostid, get; sethostid, get; sethostid, get; sethostid, get; set unique identifier of current settimers: get; set name of current host. gethostiname; setlimer: get; set value of interval timer. getitimer; get; set value of interval timer. getitimer; set, set unique identifier of current settimer, set, set unique identifier of current settimer, set, set unique identifier of current settimer, settimer set proprion. crypt(3) settlog, settl	botala, bottala, bottala, bottala,		
entry. getfsent, getfsepec, getfsfile, getfstype, settignt, endigesting, settigd, se	user and group ID, setuid,		
setuid, setuid, setuid, setuid, setuid, setgrid, setgid, setgid, set user and group ID. setuid(3) setgroups: setgroup secess list. setgroups: setders.	entry, getssent, getsspec, getssfile, getsstype,		
setgrent, getgreid, getgrnam, setgrent, endgrent; get group file entry. gethostbyname, gethostbyddr, gethostent, host. gethostent, host. gethostent, host. gethostent, sethostent; get network host entry. gethostame, getitimer, getitimer, getyset value of interval timer. getitimer, getyset value of interval timer. getitimer; getyset value of interval timer. getypose, setlon, getypose, setlon, getproneal goto. getnetent, getnetbyname, settlenethy assign buffering to a stream. getpronent, getprotobyname, getprotobyname, entry. getpwent, getprotobyname, entry. getpwent, getprotobyname, entry. getpwent, endpwent, endpendent, getpwuid, getpwam, setpwent, endpendent, getpwent, getpwould, getpwam, setpwent, endpendent, getpwould, getpwam, setpwent, endpendent, getpwent, getpwould, getpwam, setpudical, settling, consumption. getrlimit, group ID. settling, get set real and effective group in getpwent, getser velon, get			
gethostbyname, gethostbyaddr, gethostent, host. gethostid, sethostic get/set unique identifier of current sethostid(2) gethostname, getitimer,	getgrent, getgrgid, getgrnam,		
sethostid, gethostid, gethostid, gethostame, getitimer, syslog, openlog, closelog, gethetent, getnetbyname, gethostom, gethostid, gethethyname, entry, getprotobyname, entry, getpwent, getpwuid, getpwama, setpwent, entry, getpwent, getpwuid, getpwama, getpwent, getpwoid, getend, getrifictive, entry, getpwent, getpwoid, getgid, setegid, consumption, getrilimit, group ID. settrigid: set real and effective group ID. setrigid: set get get get get get get get get get g		setgroups: set group access list	setgroups(2)
gethostname, getitimer, getitimer, crypt, setbuf, setbuffer, syslog, openlog, closelog, getnetent, getnetbyandr, getprotobyname, entry. getpwent, getpwand, getpwand, getpwent, getpwand, setpwid, group ID. setuid, seteuid, getservent, getservbynent, getservent, getservbynent, getwent, getpwand, setpwid, getservent, getservbynent, getwent, getpwand, setepwid, getpwent, getpwand, setepwid, getpwent, getpwand, setepwid, getpwent, getpwand, setepwid, setepwi	gethostbyname, gethostbyaddr, gethostent,		
setitimer; get/set value of interval timer. setiouf, setbuffer, syslog, openlog, closed, getnetent, getnetbyaddr, getnetbyname, entry. getprotobynamber, getprotobyname, entry. getpwuid, getpwnid, getpwnin, getpwent, getpwuid, getpwnin, getpwnin, getpwent, getpwuid, getpwnin, getpwent, getpwnid, getpwnin, getpwnid, setridic, setral and effective group ID. setregid: set real and effective group ID. setrimit: control maximum system resource getsockopt, for changing/ random, srandom, initstate, gettimeofday, xsetroot: X window system root window parameter gettyent, gettynnin, set user and group ID. xset - X window system user gettyent, gettynnin, set user and group ID. settimit: control maximum system resource gettimeofday, setside: better random number generator; routines settimeofday; settimer; get/set and subfring to a stream. settimedic system is get process group. setprocess group. getprotocont, redprotocont; get protocol entry. getprotocont, endprotoent; get password file entry. getprotoent, endprotoent; get password file entry. setprocest group. setprotoent, endprotoent; get password file entry. setprocest group. setprotoent, endprotoent; get password file entry. setprotoent, endprotoent; get password file entry. setprotoent, endprotoent; get password file entry. setprotoent, endprotoent;			
setimp, longimp: non-local goto. setimp(s) pongimp: non-local goto. setlinebuf: assign buffering to a stream. setlogmask: control system log. setlinebuf: assign buffering to a stream. setlogmask: control system log. setprotectic, get nervety. setprotection system log. setprotection system, setprotocol entry. setprotection entry			
setbuf, setbuffer, syslog, openlog, closelog, getnetent, getnetbyaddr, getnetbyname, estelogmask: control system log. syslog(3) setnetent, getnetotopyname, getprority, getprotoent, getprotobyname, entry. getpwent, getpwuid, getpwnam, setpwent, endpwent, entry. getpwuid, getpwnam, setpwent, endpwent, getpwoid, getpwnam, setpwent, endpwent, getpwent, getpwoid, getpwnam, setpwent, endpwent, endpwent, getpwoid, getpwnam, setpwent, endpwent, endpwent, getpwoid, getpwnam, setpwent, endpwent, endpwent, setpwile: get password file entry. getpwent(3) setpwile: get password file entry. getpwent(3) setregid: set real and effective group ID. setregid: set real and effective user ID's. setrudic(2) setregid: set real and effective user ID's. setrudic(3) setrifinit: control maximum system resource gettrimit(2) setsockopt, for changing/ random, srandom, initstate, gettynent, gettynen, gettsockopt; get and set options on sockets. gettimeofday, xsetroot: X window system root window parameter gettypent, g	getitimer,		
setbuffer, syslog, openlog, closelog, getnetent, getnetbyaddr, getnetbyname, getproiotity, getprotoent, getprotobynumber, getprotobyname, entry. getpwent, endpwent, settly program suspayed file getpwent(3) setuid, setuid, setuid, setuid, setuid, setuid, group ID. setuid, setuid, getservent, getservbyport, getservbyname, getservent, getservbyport, getservbyname, getseckopt, getservent, getservent, endservent; getservent, endservent; getservent;			
syslog, openlog, closelog, gethetenth, getnethyaddr, getprotobyname, getprotobyname, entry. getpwent, getpwond, getpwam, setpwent, endpwent, setpwfile: get password file entry. getpwent, getpwent, getpwam, setpwent, endpwent, setpwfile: get password file entry. getpwent(3) setregid: set real and effective group ID. setregid: set real and effective user ID's. setregid: set user and group ID. setridis set gets and set get limit: control display setregid: set user and set limit: control display setregid: set user and set setregid: set user and set limit: control display set get set sets entry. getservent, endservent: get service entry. getservent, getsecvent, getsecvent, getsecvent, getsecvent, getsecvent, endservent: get service entry. getservent, endservent: get service entry. getsecvent, gettimeofday; gettimeofday; get; endset options on sockets. gettoday: get; endset options on sockets. gettoday: get; endset options on sockets. gettimeofday: get; endset options on sockets			
setnetent, getnetbyaddr, getnetbyname, setpgrp: set process group. getprotoent, getprotobynumber, getprotobyname, entry. getpwent, getpwuid, getpwnam, setpwent, endpwent, getpwent, getpile; get password file entry. setudota: enable/disable quotas on a file system. setuid, seteuid, seteuid, consumption. getrlimit, group ID. setrigid: set real and effective user ID's. setreuid: set real and effective user ID's. setrigid: set real and effective user ID's. setrigid: set real and effective user ID's. setrigid: set user and group ID. setrigid: set user and group ID. setricid: set real and effective user ID's. setricid: set real and effective group ID. setroid: set set real and effective group ID. setricid: set real and effective group ID. setroid: set real and effective user ID's. setricid: set real and effective user ID's. setroid: set real and effective group ID. setroid: set real and effective group ID. setroid: set real and effective user ID's. setroid: set real and effective group ID. setroid: set real and effective group ID. setroid: set real and effective user ID's. setroid: set real and effective group ID. setroid: setroid: setroid setroid: setroid			
setpgrp: set process group. getprotoent, getprotobynumber, getprotobyname, entry. getpwent, getpwuid, getpwnam, setpwent, endpwent, getprotoent, getprotoent, getprotoent, getpwuid, getpwnam, setpwent, endpwent, getpwent, getpwnam, setpwent, endpwent, setpwfile: get password file setpwfile: get password file setpwfile: get password file entry. setuid, seteuid, setruid, setgid, setgid, seteuid set real and effective group ID. setuid, setuid, seteuid, setruid, setgid, setgid: set user and group ID. setruid: set real and effective group ID. setregid: set user and group ID. setruid: set user and group ID. getservent, getservbyname, getsockopt, for changing/ random, srandom, initstate, gettimeofday, setsockopt: get and set options on sockets. gettimeofday, setting utility, setting utility, setting utility, setting utility. xsetroot: X window system root window parameter gettyent, gettynam, set user and group ID. xset - X window system user getturent, gettynam, set user and group ID. xset - X window system user gettimeofday, setting utility, setting utility, setting, telling utility, setting utility, setting, settin			
getprotoent, getprotobynumber, getpswent on a file system. getprotoent, endprotoent: get protocol entry. settprotoent, endprotoent: get protocol entry. getprotoent, endprotoent: get protocol entry. settprotoent, endprotoent: get protocol entry. getprotoent, endprotoent: get protocol entry. getprotoent, endprotoent: get protocol entry. settued: set real and effective user ID's. settregid: set real and effective user ID's. settrigid: set user and group ID. settrigid: set user and group ID. settrigid: set user and group ID. settrimit: control maximum system resource getprimit(2) settrigid: set user and group ID. settrimit: control maximum system resource getprimit(2) settrigid: set user and group ID. settrimit	genietent, genietoyadur, genietoyname,		
setprotoent, getprotobynumber, getprotobyname, entry. getpwent, getpwuid, getpwnam, setpwent, endpwent, setpwfile: get password file entry. setpwent(3) setpwent(3	getpriority.		* *
setury. getpwent, getpwnam, setpwent, endpwent, getpwent, getgid, setegid, setergid: set real and effective user ID's. setregid: set real and effective user ID's. setregid: set user and group ID. setrimit: control maximum system resource getrimit: control display screen access. setscreen(8) setservent, getservent, getservent, getservent, getservent, getservent, getservent, getservent, endservent: get service entry. getservent, endservent: get service entry. getseckopt; get and set options on sockets. getsockopt; get and set options on sockets. getsockopt(2) settime ofday; get/set date and time. settime offay: get/set date and time. settime offay: get/set date and time. gettime offay(2) setting utility. xsetroott (1) setting utility. xsetroott(1) setting utility. setting utility. xsetroott(1) set			
setpwent, getpwuid, getpwnam, setpwent, endpwent, setpwent, endpwent, getpwent, getpwent, getgid, setegid, setregid set real and effective user ID's. setuid(3) setreuid: set user and group ID. settiid, setgid, setegid; set user and settive in maximum system resource getrimit(2) setreuid: set user and settive id: set user and setuid. settivid: set user and settive id: set user and setuid: set user and settive id: set user and setuid: s			• • • • • • • • • • • • • • • • • • • •
setregid: set real and effective group ID. setreuid, seteuid, seteuid, seteuid, setreuid: set real and effective user ID's. setreuid: set real and effective user ID's. setreuid: set user and group ID. setuid, seteuid, setrimit: control maximum system resource getsockopt, setsockopt, setsockopt, setsockopt, getsockopt, getsockopt, getsockopt, getsockopt, getsockopt, setsockopt: get and set options on sockets. gettimeofday, settimeofday; settimeofday: settimeofday; settimeofd			- · · · · · · · · · · · · · · · · · · ·
setuid, seteuid, seteuid, setegid, setegid, consumption. getrlimit, group ID. setuid, seteuid, setuid, seteuid, seteuid, seteuid, seteuid, seteuid, seteuid, setuid, setuid, setgid, setegid, setrgid: set user and group ID. setuid(3) setscreen: control maximum system resource getsorkopt, getservent, getservbyparne, getsorkopt, for changing/ random, srandom, initstate, gettimeofday, setsorkopt: get and set options on sockets. getsockopt(2) random/3 set user and group ID. settimeofday; set/set date and time. gettimeofday(2) settimeofday; get/set date and time. gettimeofday(2) settimeofday: settimeofday; get/set date and time. gettimeofday(2) settimeofday; settivent, endityent: get trys file entry. settivent(3) settivent, endityent: get trys file entry. settivent(3) settivent, setuid, seteuid, setegid, setegid, setgid: setuid(3) setuid, seteuid, setrid, setgid; set user and group ID. setsockopt: get and set options on sockets. getsockopt(2) random(3) setstate: better random number generator; routines settimeofday; get/set date and time. gettimeofday(2) setting utility. settivent(3) setting utility. settivent, endityent: get trys file entry. gettivent(3) settivent, endityent: get trys file entry. gettivent(3) settivent, endityent: get legal user shells. getuid; setuid; setuid, seteuid, setegid, setegid, setegid; setuid(3) setuid, seteuid, seterid, setegid, setegid; setuid(3) setting utility. setting		setquota: enable/disable quotas on a file system	setquota(2)
setuid, setuid, setgid, setegid, consumption. getrlimit, group ID. setuid, setuid, setgid, setegid, setgid; set user and group ID. setuid, setuid, setgid, setgid; set user and group ID. setuid, setgid, setgid; set user and group ID. setuid(3) setsimit: control maximum system resource getrlimit(2) setruid, setgid, setgid, setgid; set user and group ID. setuid(3) setsicreen: control display screen access. setscreen(8) setscreen: control display screen access. setscreen(access. setscreen(access. setscreen(access. setscreen: control display screen access. setscreen(access. setscreen: control display screen access. setscreen: control display screen access. setscreen(access. setscreen: control display screen access. setter control maximum system resource access. setured: setured: setured: setured: setured: setured: acceletate better random number generator; routines gettimeofday: gettimeofd			setregid(2)
consumption. getrlimit, group ID. setuid, seteuid, setgid, setgid, setgid, setgid, set get user and setuid(3) setscreen; control display screen access. setscreen(3N) getservent, getservbyport, getservbypname, getsockopt, for changing/ random, srandom, initstate, gettimeofday, xsetroot: X window system root window parameter getttyent, getttynam, set user and group ID. xset - X window system user getusershell, continue, cd, eval, exec, exit, export, login,/ xstr: extract strings from C programs to implement exit: leave system: issue a csh: a eval: re-evaluate poop pushd: push alias: consumption. gettlimit; control maximum system resource setrid(3) setugid, setegid, setgid; set user and set options on sockets. getsockopt(2) setsockopt: get and set options on sockets. getsockopt(2) setstate: better random number generator; routines setstimeofday; get/set date and time. gettimeofday(2) setting utility. xsettimeofday: gettyent, gettityent, endityent: get ttys file entry. gettityent(3) setuid, seteuid, seteuid, setgid, setgid, setgid; setgid: setuid(3) setup program. xset(1) settivent, endityent: get ttys file entry. gettivent(3) setuid, seteuid, setgid, setgid, setgid, setgid; setuid(3) setup program. xset(1) settivent, endityent: get ttys file entry. gettivent(3) setuid, setuid, setgid, setgid			
getservent, getservbyname, getsockopt, for changing/ random, srandom, initstate, gettimeofday, xsetroot: X window system root window parameter gettyent, gettynam, set user and group ID. xset - X window system user getusershell, continue, cd, eval, exec, exit, export, login,/ xstr: extract strings from C programs to implement exit: leave system: issue a eval: re-evaluate popd: pop pushd: push alias: group ID. setuid, setgid, setegid; set user and setservent(3N) setsockopt: get and set options on sockets. getservent, endservent: get service entry. setsockopt: get and set options on sockets. getservent, endservent: get service entry. setservent, endservent: get service entry. setservent, endservent: get service entry. setservent, endservent: get service entry. setsockopt: get and set options on sockets. setservent, endservent: get service entry. setservent entry. setserveit entry. setservententry. setserveit entry. setservententry. setserveit entry. setserveit entr			
getservent, getservbyport, getservbyname, getsockopt, for changing/ random, srandom, initstate, gettimeofday, sestockopt: get and set options on sockets. getsockopt(2) setsockopt: get and set options on sockets. getsockopt(2) setsockopt: get and set options on sockets. getsockopt(2) setsockopt: get and set options on sockets. getsockopt(2) settimeofday; get/set date and time. gettimeofday(2) settimeofday: get/set date and time. gettimeofday(2) setting utility. setting program. setusershell, endusershell: get legal user shells. getusershell(3) setting program. settings from C programs to implement exit: leave system: issue a csh: a eval: re-evaluate popd: pop pushd: push alias:	consumption. getriimit,		
getservent, getservbyport, getservbyname, getsockopt, for changing/ random, srandom, initstate, gettimeofday, gettimeofday, setstate: better random number generator; routines gettimeofday; get/set date and time. gettimeofday(2) setting utility setstate: better random number generator; routines gettimeofday(2) setting utility setting setting utility setting setting utility setting setting setting utility setting setting utility setting setting utility setting utility setting utility setting setting utility	group ID. setula, seteula,		\ <i>,</i>
getsockopt, for changing/ random, srandom, initstate, gettimeofday, setstate: better random number generator; routines gettimeofday, settimeofday; get/set date and time. gettimeofday(2) settimeofday: get/set date and time. gettimeofday(2) setting utility. setting setting utility. setting setting utility. setting setting utility. setting setting utility. setting setting utility.	cotconyont cotconybynant cotconybynama		` '
for changing/ random, srandom, initstate, gettimeofday, settimeofday; get/set date and time. gettimeofday(2) settimeofday: get/set date and time. gettimeofday(2) settimeofday: get/set date and time. gettimeofday(2) setting utility. setting util			. ,
gettimeofday, xsetroot: X window system root window parameter getttyent, getttynam, set user and group ID. xset - X window system user getusershell, continue, cd, eval, exec, exit, export, login,/ xstr: extract strings from C programs to implement exit: leave system: issue a csh: a eval: re-evaluate popd: pop pushd: push alias: gettimeofday: get/set date and time.			
xsetroot: X window system root window parameter getttyent, getttynam, set user and group ID. xset - X window system user getusershell, continue, cd, eval, exec, exit, export, login,/ xstr: extract strings from C programs to implement exit: leave system: issue a csh: a eval: re-evaluate popd: pop pushd: push alias: xset user and group ID. settiyent, endttyent: get ttys file entry. gettyent(3) setuid, seteuid, setgid, setgid; setgid: setuid(3) setuid, setuid, setgid, setgid; setgid: setuid(3) setuid, setuid, setgid, setgid; setgid: setuid(3) setuid, setgid, setgid, setgid: setgid: setuid(3) setuid, setgid, setgid, setgid: setgid: setgid(3) setgid, setgid:			
getttyent, getttynam, set user and group ID. xset - X window system user getusershell, continue, cd, eval, exec, exit, export, login,/ xstr: extract strings from C programs to implement exit: leave system: issue a csh: a eval: re-evaluate popd: pop pushd: push alias: getttyent, endttyent: get ttys file entry			
set user and group ID. xset - X window system user getusershell, endusershell: get legal user shells. continue, cd, eval, exec, exit, export, login,/ xstr: extract strings from C programs to implement exit: leave system: issue a csh: a eval: re-evaluate popd: pop pushd: push set user and group ID. setuid, seteuid, setegid, setegid: xset(1) setup program xset(1) setusershell, endusershell: get legal user shells sh(1) shell command xstr(1) shell command xstr(1) shell command interpreter) with C-like syntax csh(1) shell data csh(1) shell directory stack csh(1)			
xset - X window system user getusershell, setusershell, endusershell: get legal user shells. getusershell(3) sh, for, case, if, while, :, . break, sh(1) shared strings from C programs to implement exit: leave system: issue a shell command	set user and group ID.		
getusershell, continue, cd, eval, exec, exit, export, login,/ xstr: extract strings from C programs to implement exit: leave system: issue a csh: a eval: re-evaluate popd: pop pushd; push dias: Setusershell, endusershell: get legal user shells			
continue, cd, eval, exec, exit, export, login,/ xstr: extract strings from C programs to implement exit: leave system: issue a csh: a eval: re-evaluate popd: pop pushd: push alias: sh, for, case, if, while, :, ., break, sh(1) shared strings xstr(1) shared strings csh(1) shared strings csh(1) shell command system(3) csh: a shell data csh(1) shell directory stack csh(1) shell directory stack csh(1) shell directory stack csh(1) shell macros csh(1)			
xstr: extract strings from C programs to implement exit: leave system: issue a shell command	continue, cd, eval, exec, exit, export, login,/	sh, for, case, if, while, :, ., break,	
exit: leave shell	xstr: extract strings from C programs to implement	shared strings	1 1
system: issue a shell command			
eval: re-evaluate popd: pop shell directory stack.	system: issue a	shell command.	* *
popd: pop shell directory stack			csh(1)
pushd: push shell directory stack	_	shell data	
alias: shell macros			
suspend suspend a snell, resuming its superior			
	suspena: suspena a	such, resulting its superior	can(1)

rshd: remote	shell server	rshd(8C)
set: change value of	shell variable	csh(1)
@: arithmetic on	shell variables	csh(1)
unset: discard	shell variables	csh(1)
exec: overlay	shell with specified command	csh(1)
setusershell, endusershell: get legal user	shells. getusershell,	getusershell(3)
	shift: manipulate argument list.	csh(1)
/exec, exit, export, login, read, readonly, set,	shift, times, trap, umask, wait: command language	sh(l)
long,	short: integer object conversion.	long(3F)
server. rvdshow:	show connections to Remote Virtual Disk (RVD)	rvdshow(8)
		, ,
groups:		groups(1)
uptime:	show how long system has been up	uptime(1)
lastcomm:	show last commands executed in reverse order	lastcomm(1)
netstat:	show network status	netstat(1)
uusnap:	show snapshot of the UUCP system	uusnap(8C)
construct a file. what:	show what versions of object modules were used to	what(1)
shutdown:	shut down part of a full-duplex connection	shutdown(2)
	shutdown: close down the system at a given time	shutdown(8)
rvdshut: force	shutdown of Remote Virtual Disk (RVD) server	rvdshut(8)
connection.	shutdown: shut down part of a full-duplex	shutdown(2)
	sigblock: block signals.	sigblock(2)
calls.	siginterrupt: allow signals to interrupt system	siginterrupt(3)
login:	sign on.	login(1)
•		
sigreturn: return from	signal.	sigreturn(2)
pause: stop until	signal	pause(3C)
signal: change the action for a	signal.	signal(3F)
alarm: schedule	signal after specified time.	alarm(3C)
ualarm: schedule	signal after specified time.	ualarm(3)
	signal: change the action for a signal	signal(3F)
sigvec: software	signal facilities.	sigvec(2)
sigvec: software	signal facilities.	sigvec(2)
signal: simplified software	signal facilities.	signal(3C)
sigsetmask: set current	signal mask.	sigsetmask(2)
psignal, sys siglist: system	signal messages.	psignal(3)
paignai, aya_aignat. ayatem	signal: simplified software signal facilities.	signal(3C)
signtants and and for out		
sigstack: set and/or get	signal stack context	sigstack(2)
kill: send	signal to a process	kill(2)
kill: send a	signal to a process	kill(3F)
killpg: send	signal to a process group.	killpg(2)
sigblock: block	signals	sigblock(2)
sigpause: atomically release blocked	signals and wait for interrupt	sigpause(2)
(temporary). infnan:	signals invalid floating-point operations on a VAX	infnan(3M)
siginterrupt: allow	signals to interrupt system calls	siginterrupt(3)
wait for interrupt.	signause: atomically release blocked signals and	sigpause(2)
	sigreturn: return from signal.	sigreturn(2)
	sigsetmask: set current signal mask.	sigsetmask(2)
	sigstack: set and/or get signal stack context	sigstack(2)
		sigvec(2)
	sigvec: software signal facilities.	
	sigvec: software signal facilities.	sigvec(2)
	simplified software signal facilities.	signal(3C)
tc: photoypesetter	simulator	tc(1)
trigonometric functions and their inverses.	sin, cos, tan, asin, acos, atan, atan2:	sin(3M)
	sinh, cosh, tanh: hyperbolic functions	sinh(3M)
null: data	sink	null(4)
uupoll: poll a remote UUCP	site	uupoll(8C)
brk, sbrk: change data segment	size	brk(2)
getdtablesize: get descriptor table	size	getdtablesize(2)
getpagesize: get system page	size	getpagesize(2)
pagesize: print system page	size	pagesize(1)
size:	size of an object file.	size(1)
SIZC.	size size of an object file.	size(1)
dialements coloniate deCoult dialementition	· ·	
diskpart: calculate default disk partition	sizes	diskpart(8)
diskpart: calculate default disk partition	sizes	diskpart(8)
interfaces.		slattach(8C)
	slattach: attach serial lines as network	
	sleep: suspend execution for an interval	sleep(1)
	sleep: suspend execution for an interval sleep: suspend execution for an interval	sleep(3F)
	sleep: suspend execution for an interval sleep: suspend execution for an interval sleep: suspend execution for interval	
sc: IBM 9332 disks using the IBM	sleep: suspend execution for an interval sleep: suspend execution for an interval	sleep(3F)
sc: IBM 9332 disks using the IBM spline: interpolate	sleep: suspend execution for an interval sleep: suspend execution for an interval sleep: suspend execution for interval	sleep(3F) sleep(3) sc(4)
spline: interpolate	sleep: suspend execution for an interval sleep: suspend execution for an interval sleep: suspend execution for interval	sleep(3F) sleep(3) sc(4) spline(1G)
spline: interpolate uusnap: show	sleep: suspend execution for an interval. sleep: suspend execution for an interval. sleep: suspend execution for interval. Small Computer System Interface (SCSI) Adapter. smooth curve. snapshot of the UUCP system.	sleep(3F) sleep(3) sc(4) spline(1G) uusnap(8C)
spline: interpolate uusnap: show accept: accept a connection on a	sleep: suspend execution for an interval. sleep: suspend execution for an interval. sleep: suspend execution for interval. Small Computer System Interface (SCSI) Adapter. smooth curve. snapshot of the UUCP system. socket.	sleep(3F) sleep(3) sc(4) spline(1G) uusnap(8C) accept(2)
spline: interpolate uusnap: show accept: accept a connection on a bind: bind a name to a	sleep: suspend execution for an interval. sleep: suspend execution for an interval. sleep: suspend execution for interval. Small Computer System Interface (SCSI) Adapter. smooth curve. snapshot of the UUCP system. socket. socket.	sleep(3F) sleep(3) sc(4) spline(1G) uusnap(8C) accept(2) bind(2)
spline: interpolate uusnap: show accept: accept a connection on a bind: bind a name to a connect: initiate a connection on a	sleep: suspend execution for an interval. sleep: suspend execution for an interval. sleep: suspend execution for interval. Small Computer System Interface (SCSI) Adapter. smooth curve. snapshot of the UUCP system. socket. socket.	sleep(3F) sleep(3) sc(4) spline(1G) uusnap(8C) accept(2) bind(2) connect(2)
spline: interpolate uusnap: show accept: accept a connection on a bind: bind a name to a connect: initiate a connection on a listen: listen for connections on a	sleep: suspend execution for an interval. sleep: suspend execution for an interval. sleep: suspend execution for interval. Small Computer System Interface (SCSI) Adapter. smooth curve. snapshot of the UUCP system. socket. socket. socket.	sleep(3F) sleep(3) sc(4) spline(1G) uusnap(8C) accept(2) bind(2) connect(2) listen(2)
spline: interpolate uusnap: show accept: accept a connection on a bind: bind a name to a connect: initiate a connection on a listen: listen for connections on a cv, recvfrom, recvmsg: receive a message from a	sleep: suspend execution for an interval. sleep: suspend execution for an interval. sleep: suspend execution for interval. Small Computer System Interface (SCSI) Adapter. smooth curve. snapshot of the UUCP system. socket. socket. socket. socket.	sleep(3F) sleep(3) sc(4) spline(1G) uusnap(8C) accept(2) bind(2) connect(2) listen(2) recv(2)
spline: interpolate uusnap: show accept: accept a connection on a bind: bind a name to a connect: initiate a connection on a listen: listen for connections on a	sleep: suspend execution for an interval. sleep: suspend execution for an interval. sleep: suspend execution for interval. Small Computer System Interface (SCSI) Adapter. smooth curve. snapshot of the UUCP system. socket. socket. socket.	sleep(3F) sleep(3) sc(4) spline(1G) uusnap(8C) accept(2) bind(2) connect(2) listen(2)

imm: IMD row.	socket: create an endpoint for communication socket interface	socket(2)
imp: IMP raw getsockname: get	socket interface	imp(4P) getsockname(2)
gg	socketpair: create a pair of connected sockets	socketpair(2)
getsockopt, setsockopt: get and set options on	sockets.	getsockopt(2)
socketpair: create a pair of connected	sockets. soelim: eliminate .so's from nroff input.	socketpair(2) soelim(1)
lo:	software loopback network interface	lo(4)
in ip packets nsip:	software network interface encapsulating ns packets .	nsip(4)
sigvec:	software signal facilities.	sigvec(2)
sigvec:	software signal facilities.	sigvec(2)
signal: simplified support: hardware and	software signal facilities	signal(3C) support(1)
tsort: topological	sort	tsort(1)
qsort: quicker	sort	qsort(3)
qsort: quick	sort	qsort(3F)
sortbib: sort:	sort bibliographic database	sortbib(1) sort(1)
501 t.	sort: sort or merge files.	sort(1)
	sortbib: sort bibliographic database.	sortbib(1)
comm: select or reject lines common to two	sorted files.	comm(1)
look: find lines in a	sorted list.	look(1)
soelim: eliminate soelim: eliminate .	so's from nroff input	soelim(1) soelim(1)
indent: indent and format C program	source	indent(1)
mkstr: create an error message file by massaging C	source.	mkstr(1)
whereis: locate	source, binary, and or manual for program	whereis(1)
Para to the control of the territory of	source: read commands from file.	csh(1)
line, circle, arc, move, cont, point, linemod, expand, unexpand: expand tabs to	space, closepl: graphics interface. /erase, label, spaces, and vice versa	plot(3X)
expand, unexpand: expand tabs to way. vfork:	spaces, and vice versa	expand(1) vfork(2)
, a, , , , , , , , , , , , , , , , , ,	speaker: console speaker interface	speaker(4)
speaker: console	speaker interface	speaker(4)
exec: overlay shell with	specified command.	csh(1)
truncate, ftruncate: truncate a file to a alarm: schedule signal after	specified length	truncate(2) alarm(3C)
alarm: execute a subroutine after a	specified time.	alarm(3F)
ualarm: schedule signal after	specified time.	ualarm(3)
swapon:	specify additional device for paging and swapping	swapon(8)
anall	spell, spellin, spellout: find spelling errors spellin, spellout: find spelling errors	spell(1) spell(1)
spell, spell, spellin, spellout: find	spelling errors.	spell(1)
spell, spellin,	spellout: find spelling errors.	spell(1)
vdabort: abort and	spin down a drive	vdabort(8)
vdspin, vdspind: spin up or	spin down a Remote Virtual Disk (RVD)	vdspin(2)
vdspin, vdspind: spinup, spindown:	spin up or spin down a Remote Virtual Disk (RVD) spin up/down Remote Virtual Disk (RVD) pack	vdspin(2) spinup(8)
spinap, spinaowii. rvdflush:	spindown client's Remote Virtual Disk (RVD) packs	rvdflush(8)
rvddown: force	spindown of a Remote Virtual Disk (RVD) pack	rvddown(8)
pack. spinup,	spindown: spin up/down Remote Virtual Disk (RVD) .	spinup(8)
(RVD) pack.	spinup, spindown: spin up/down Remote Virtual Disk .	spinup(8)
split:	spline: interpolate smooth curve	spline(1G) split(1)
files. fsplit:	split a multi-routine Fortran file into individual	fsplit(1)
frexp, ldexp, modf:	split into mantissa and exponent.	frexp(3)
• • •	split: split a file into pieces	split(1)
uuclean: uucp	spool directory clean-up.	uuclean(8C)
lpq: atrm: remove jobs	spool queue examination program	lpq(1) atrm(1)
lprm: remove jobs from the line printer	spooling queue.	lprm(1)
Pageprinter. ppt:	spooling system filter for the IBM 3812	ppt(8)
	spp: Xerox Sequenced Packet Protocol	spp(4P)
printf, fprintf,		printf(3S)
cbrt, cbrt, sqrt: cube root,	sqrt: cube root, square root	sqrt(3M) sqrt(3M)
rand.	srand: random number generator.	rand(3C)
generator; routines for changing/ random,	srandom, initstate, setstate: better random number	random(3)
scanf, fscanf,	sscanf: formatted input conversion	scanf(3S)
	st: streaming-tape interface	st(4) stab(5)
popd: pop shell directory	stack	csh(1)
pushd: push shell directory	stack.	csh(1)
sigstack: set and/or get signal	stack context.	sigstack(2)
drtest:	standalone disk test program.	drtest(8)
sautil: bad144: read/write dec	standalone utility package standard 144 bad sector information	sautil(8r) bad144(8)
ondian. Tead, write dec	CONTRACTOR OF THE PROPERTY OF	Jun 1-7-(U)

stdio:		
		lio(3S)
htable: convert NIC		able(8)
ddn: DDN		n(4)
stdemul:		lemul(4)
rc.config: configuration file for	startup scripts rc.	config(5)
		ıt(2)
	stat, Istat, fstat: get file status sta	t(3F)
return machine and process floating point	state. getfloatstate: get	tfloatstate(2)
stty, gtty: set and get terminal		y(3C)
fsync: synchronize a file's in-core		/nc(2)
if: conditional		n(1)
fstab:		ab(5)
tailor: work	· · · · · · · · · · · · · · · · · · ·	lor(8)
hashstat: print command hashing		n(1)
iostat: report I/O		* *
cause Remote Virtual Disk (RVD) server to log		tat(1)
` / '		dlog(8)
vdstats: acquire client Remote Virtual Disk (RVD)		stats(2)
vdstats: list client Remote Virtual Disk (RVD)		stats(8)
vmstat: report virtual memory		istat(1)
vmstat: report virtual memory		istat(1)
systat: display system		stat(1)
netstat: show network	status net	lstat(1)
ps: process	status	(1)
stat, Istat, fstat: get file		it(2)
exit: terminate process with	_	it(3F)
stat, Istat, fstat: get file		ıt(3F)
printer3812: IBM 3812 Pageprinter		inter3812(5)
ferror, feof, clearerr, fileno: stream		ror(3S)
sysline: display system status on		sline(1)
sysline: display system	· · · · · · · · · · · · · · · · · · ·	sline(1)
rwhod: system	•	hod(8C)
i whoa. system		
		lemul(4)
		lio(3S)
directories.		cky(8)
		1(1)
halt:		lt(8)
halt:		lt(8)
pause:	stop until signal	use(3C)
icheck: file system	storage consistency check ich	eck(8)
up: unibus	storage module controller/drives up((4)
subroutines. dbminit, fetch,	store, delete, firstkey, nextkey: data base dbr	m(3X)
strien, index, rindex: string operations.		ing(3)
strlen, index, rindex: string operations. rindex: string operations, streat, strncat.	streat, strneat, stremp, strnemp, strepy, strnepy, stri	ing(3) ing(3)
rindex: string operations. strcat, strncat,	streat, strncat, stremp, strnemp, strepy, strnepy, strnepy, strnemp, stremp, strnemp, strepy, strnepy, strlen, index, stri	ing(3)
rindex: string operations. strcat, strncat, operations. strcat, strncat, strcmp, strncmp,	streat, strneat, stremp, strnemp, strepy, strnepy, strnemp, stremp, stremp, strepy, strnepy, strlen, index, strepy, strnepy, strlen, index, rindex: string stri	ing(3) ing(3)
rindex: string operations. strcat, strncat, operations. strcat, strncat, strcmp, strncmp, fclose, fflush: close or flush a	streat, strneat, stremp, strnemp, strepy, strnepy, stremp, strnemp, strepy, strnepy, strlen, index, strepy, strnepy, strlen, index, rindex: string stream	ing(3) ing(3) ose(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a	streat, strneat, stremp, strnemp, strepy, strnepy, stremp, strnemp, strepy, strnepy, strlen, index, string strepy, strnepy, strlen, index, rindex: string stream	ing(3) ing(3) ose(3S) oen(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a	streat, strncat, stremp, strncmp, strepy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string stream	ing(3) ing(3) ose(3S) oen(3S) ek(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from	streat, strncat, stremp, strncmp, strepy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string stream	ing(3) ing(3) ose(3S) oen(3S) ek(3S) cc(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a	strcat, strncat, strcmp, strncmp, strcpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string stream	ing(3) ing(3) ose(3S) oen(3S) ek(3S) ic(3S) is(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a	streat, strneat, stremp, strnemp, strepy, strnepy, stremp, strnemp, strepy, strnepy, strlen, index, string strepy, strnepy, strlen, index, rindex: string stream	ing(3) ing(3) ing(3) ose(3S) ose(3S) oek(3S) ic(3S) ic(3S) ic(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a	streat, strneat, stremp, strnemp, strepy, strnepy, stremp, strnemp, strepy, strnepy, strlen, index, string strepy, strnepy, strlen, index, rindex: string stream. st	ing(3) ing(3) ing(3) ose(3S) ose(3S) eek(3S) ic(3S) ic(3S) tc(3S) ts(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a	strcat, strncat, strcmp, strncmp, strcpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string stricteam. stream.	ing(3) ing(3) ose(3S) ose(3S) oen(3S) ek(3S) is(3S) is(3S) tc(3S) ts(3S) buf(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input	strcat, strncat, strcmp, strncmp, strcpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string stricteam. stream.	ing(3) ing(3) ose(3S) ose(3S) oen(3S) ek(3S) ic(3S) ic(3S) tc(3S) ts(3S) buf(3S) getc(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed:	strcat, strncat, strcmp, strncmp, strcpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string stricteam. follows stream. fop stream. stream. getc, stream. stream. getc, stream. stream. getc, stream. stream. getc, stream. setbuf, setdiam.	ing(3) ing(3) ing(3) ose(3S) oen(3S) ek(3S) ic(3S) ic(3S) it(3S) it(3S) obuf(3S) getc(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freepen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno:	strcat, strncat, strcmp, strncmp, strcpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string string stream	ing(3) ing(3) ing(3) ose(3S) oen(3S) ek(3S) ic(3S) is(3S) to(3S) buf(3S) buf(3S) getc(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freepen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: rcmd, rresvport, ruserok: routines for returning a	strcat, strncat, strcmp, strncmp, strncpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, stristrcpy, strncpy, strlen, index, rindex: string stristream	ing(3) ing(3) ing(3) ose(3S) oen(3S) ek(3S) ic(3S) ic(3S) it(3S) to(3S) buf(3S) getc(3S) if(1) ror(3S) ind(3)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return	strcat, strncat, strcmp, strncmp, strcpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string string stream.	ing(3) ing(3) ing(3) ose(3S) ose(3S) oen(3S) ic(3S) ic(3S) it(3S) it(3S) it(3S) it(3S) it(3S) it(1) if(1) if(1) it(3S) it(3S) it(1) it(3S) it(3S) it(1) it(3S) it(1) it(3S) it(3S) it(1) it(3S) it(1) it(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control	strcat, strncat, strcmp, strncmp, strncpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string string stream. str	ing(3) ing(3) ing(3) ose(3S) ose(3S) ek(3S) ic(3S) is(3S) tc(3S) ts(3S) buf(3S) getc(3S) i(1) ror(3S) md(3) iec(3)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return	strcat, strncat, strcmp, strncmp, strcpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string stricpy, strncpy, strlen, index, rindex: string strcam. stream. stream to a remote command. stream to a remote command. stream to a remote virtual Disk (RVD) server. rvd streaming tape buffered read.	ing(3) ing(3) ing(3) ose(3S) ose(3S) oen(3S) ic(3S) ic(3S) it(3S) it(3S) it(3S) it(3S) it(3S) it(1) if(1) if(1) it(3S) it(3S) it(1) it(3S) it(3S) it(1) it(3S) it(1) it(3S) it(3S) it(1) it(3S) it(1) it(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control	strcat, strncat, strcmp, strncmp, strncpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string string stream. str	ing(3) ing(3) ing(3) ose(3S) ose(3S) oen(3S) ic(3S) ic(3S) tc(3S) ts(3S) buf(3S) getc(3S) i(1) ror(3S) md(3) isec(3) isec(3)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer:	strcat, strncat, strcmp, strncmp, strncpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string string stream.	ing(3) ing(3) ing(3) ose(3S) ose(3S) oek(3S) ic(3S) is(3S) is(3S) buf(3S) getc(3S) i(1) ror(3S) ror(3S) idec(3) idend(8) iffer(8) 4) ite(3F)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st:	strcat, strncat, strcmp, strncmp, strncpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string string stream.	ing(3) ing(3) ing(3) ose(3S) ose(3S) oek(3S) ic(3S) is(3S) is(3S) buf(3S) getc(3S) i(1) ror(3S) ror(3S) idec(3) idend(8) iffer(8) 4) ite(3F)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII	strcat, strncat, strcmp, strncmp, strncpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcpy, strncpy, strlen, index, rindex: string string stream.	ing(3) ing(3) ing(3) ose(3S) ose(3S) oek(3S) ic(3S) is(3S) it(3S) it(3S) getc(3S) i(1) ror(3S) ror(3S) idsend(8) iffer(8) 4) ing(3G)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a	strcat, strncat, strcmp, strncmp, strncpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcey, strncpy, strlen, index, rindex: string stream. stream to a remote command. stream to a remote command. stream to Remote Virtual Disk (RVD) server. streaming tape buffered read. streaming-tape interface. string. string. string.	ing(3) ing(3) ing(3) ose(3S) ose(3S) oek(3S) ek(3S) tc(3S) ts(3S) buf(3S) getc(3S) if(1) ror(3S) nd(3) tec(3) isend(8) uffer(8) 4) tte(3F) ing(3G) is(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a	strcat, strncat, strcmp, strncmp, strncpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string stristrcpy, strncpy, strlen, index, rindex: string stristream.	ing(3) ing(3) ing(3) ose(3S) ose(3S) oen(3S) ek(3S) is(3S) tc(3S) ts(3S) buf(3S) getc(3S) if(1) ror(3S) nd(3) isec(3) isend(8) iffer(8) 4) itte(3F) ing(3G) is(3S) ts(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a bcopy, bcmp, bzero, ffs: bit and byte	strcat, strncat, strcmp, strncmp, strncpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcey, strncpy, strlen, index, rindex: string string stream. stream to a remote command. stream to a remote virtual Disk (RVD) server. streaming tape buffered read. string. string. string. string. string. string from a stream. string operations. bst.	ing(3) ing(3) ing(3) ose(3S) ose(3S) ose(3S) ic(3S) ic(3S) its(3S) buf(3S) getc(3S) getc(3S) if(1) ror(3S) md(3) itsec(3) disend(8) iffer(8) 4) itse(3F) its(3S) its(3S) its(3S) its(3S) its(3S) its(3S) itse(3S) itse(3S) itse(3S) itse(3S) itse(3S) itse(3S) itse(3S) itse(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a bcopy, bcmp, bzero, ffs: bit and byte strnemp, strepy, strnepy, strlen, index, rindex:	strcat, strncat, strcmp, strncmp, strncpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string strcam, strcpy, strlen, index, rindex: string string stream. stream outc, stream. stream. stream. stream. stream. stream. stream. stream on a remote command. stream to a remote command. stream to a remote command. stream to a remote virtual Disk (RVD) server. streaming tape buffered read. string. string. string. string. string from a stream. string operations. string operations. string operations. string string string string operations. string string. string operations. string string string. string operations. string string. string operations.	ing(3) ing(3) ing(3) ing(3) ose(3S) ose(3S) oen(3S) ic(3S) ic(3S) its(3S) buf(3S) getc(3S) if(1) ror(3S) ind(3) itsend(8) iffer(8) 4) itte(3F) iting(3G) its(3S) its(3S) its(3S) its(3S) its(3S) its(3S) its(3S) its(3S)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a bcopy, bcmp, bzero, ffs: bit and byte strnemp, stropy, strnepy, strlen, index, rindex: extract strings from C programs to implement shared	strcat, strncat, strcmp, strncmp, strncpy, stricmp, strncmp, strcpy, strncpy, strncp	ing(3) ing(3) ing(3) ose(3S) ose(3S) ose(3S) ek(3S) is(3S) is(3S) is(3S) buf(3S) getc(3S) f(1) ror(3S) md(3) iec(3) dsend(8) iffer(8) 4) ing(3G) is(3S) is(3
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI String: draw a gets, fgets: get a puts, fputs: put a bcopy, bcmp, bzero, ffs: bit and byte strnemp, strepy, strnepy, strlen, index, rindex: extract strings from C programs to implement shared other binary, file.	strcat, strncat, strcmp, strncmp, strncpy, strncpy, strcmp, strncmp, strcpy, strncpy, strlen, index, string stream. stream editor. stream to a remote command. stream to a remote command. stream to Remote Virtual Disk (RVD) server. streaming tape buffered read. streaming-tape interface. string. string. string. string operations. string operations. string operations. strings: strings: string string the printable strings in a object, or	ing(3) ing(3) ing(3) ose(3S) ose(3S) ose(3S) ek(3S) is(3S) is(3S) is(3S) buf(3S) getc(3S) f(1) ror(3S) md(3) is(c(3S) dec(3S) iffer(8) 4) ing(3G) is(3S) iring(3) iring(3) iring(3) iring(3) iring(3) iring(3) iring(1)
rindex: string operations. streat, strncat, operations. streat, strncat, stremp, strncmp, felose, fillush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a bcopy, bcmp, bzero, ffs: bit and byte strncmp, strcpy, strncpy, strlen, index, rindex: extract strings from C programs to implement shared other binary, file. strings. xstr: extract	strcat, strncat, strcmp, strncmp, strncpy, stricmp, strncmp, strcpy, strncpy, strlen, index, strictical strictical strictical strictical stream. stream editor. stream to a remote command. stream to a remote command. stream to Remote Virtual Disk (RVD) server. streaming tape buffered read. string. strings.	ing(3) ing(3) ing(3) ing(3) ose(3S) ose(3S) oen(3S) ek(3S) ic(3S) its(3S) buf(3S) getc(3S) if(1) ror(3S) nd(3) isec(3) isend(8) iffer(8) 4) itte(3F) ing(3G) its(3S) itring(3) iring(3) irr(1) ings(1) irr(1)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a bcopy, bcmp, bzero, ffs: bit and byte strnemp, strepy, strnepy, strlen, index, rindex: extract strings: find the printable	strcat, strncat, strcmp, strncmp, strncpy, stricmp, strncmp, strcpy, strncpy, strlen, index, strictical strict	ing(3) ing(3) ing(3) ing(3) ose(3S) ose(3S) oen(3S) ek(3S) ic(3S) its(3S) buf(3S) getc(3S) if(1) ror(3S) md(3) icec(3) isend(8) iffer(8) 44) ing(3G) is(3S) its(3S) itring(3) iring(3) iring(3) irr(1) ings(1)
rindex: string operations. streat, strncat, operations. streat, strncat, stremp, strncmp, felose, fillush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a bcopy, bcmp, bzero, ffs: bit and byte strncmp, strcpy, strncpy, strlen, index, rindex: extract strings from C programs to implement shared other binary, file. strings. xstr: extract	strcat, strncat, strcmp, strncmp, strncpy, stricmp, strncmp, strcpy, strncpy, strlen, index, strictical strictical stream. Strictical stream. Stream status inquiries. Set stream to a remote command. Fertical stream to a remote command. Fertical stream to Remote Virtual Disk (RVD) server. From streaming tape buffered read. Streaming. String. String. String. String. String. String. String string operations. Streat, strncat, strcmp, stristing operations. Strings from C programs to implement shared strings in a object, or other binary, file. Stristing filename affixes. Sassing is string in a object, or other binary, file. Stristing filename affixes.	ing(3) ing(3) ing(3) ing(3) ose(3S) ose(3S) oen(3S) ek(3S) ic(3S) tc(3S) ts(3S) buf(3S) getc(3S) if(1) ror(3S) nd(3) dec(3) dsend(8) uffer(8) 4) tte(3F) ting(3G) is(3S) tring(3) tri(1) ings(1) ings(1) sename(1)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a beopy, bemp, bzero, ffs: bit and byte strnemp, strepy, strnepy, strlen, index, rindex: extract strings from C programs to implement shared other binary, file. strings. xstr: extract strings: find the printable basename:	strcat, strncat, strcmp, strncmp, strncpy, stricmp, strncmp, strcpy, strncpy, strlen, index, string stream. stream to a remote command. stream to a remote virtual Disk (RVD) server. streaming tape buffered read. string. string. string. string. string from a stream. string operations. string operations. string string string the printable strings in a object, or strings from C programs to implement shared strip illename affixes. strip: remove symbols and relocation bits.	ing(3) ing(3) ing(3) ing(3) ose(3S) ose(3S) ose(3S) ic(3S) ic(3S) its(3S) buf(3S) getc(3S) if(1) ror(3S) ind(3) itsec(3) idsend(8) iffer(8) 4) ite(3F) iting(3G) iting(3) itring(3) itring(3) itring(1) itings(1) isename(1) ip(1)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fleno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a bcopy, bcmp, bzero, ffs: bit and byte strncmp, strcpy, strncpy, strlen, index, rindex: extract strings from C programs to implement shared other binary, file. strings: strings: strings. xstr: extract strings: find the printable basename:	strcat, strncat, strcmp, strncmp, strncpy, stricmp, strncmp, strcpy, strncpy, strlen, index, striction, strcpy, strlen, index, rindex: string stream. stream to a remote command. stream to a remote virtual Disk (RVD) server. streaming tape buffered read. streaming-tape interface. string. string. string. string. string. string from a stream. string operations. string operations. string string operations. strings from C programs to implement shared strings in a object, or other binary, file. strip filename affixes. strip remove symbols and relocation bits. stristrlen, index, rindex: string operations.	ing(3) ing(3) ing(3) ing(3) ose(3S) ose(3S) ose(3S) ic(3S) ic(3S) its(3S) buf(3S) getc(3S) if(1) ror(3S) ind(3) itsend(8) iffer(8) 4) itsend(3G) its(3S) its(3S) itring(3) itring(3) itring(3) itring(1) itring(1) itring(1) itring(1) itring(3)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a bcopy, bcmp, bzero, ffs: bit and byte strnemp, strepy, strnepy, strlen, index, rindex: extract strings from C programs to implement shared other binary, file. strings: find the printable basename: streat, strneat, stremp, strnemp, strepy, strnepy, index, rindex: string operations. streat,	strcat, strncat, strcmp, strncmp, strncpy, stricmp, strncmp, strcpy, strncpy, strlen, index, strictical strcpy, strncpy, strlen, index, rindex: string strictical stream. stream editor. stream to a remote command. stream to a remote command. stream to Remote Virtual Disk (RVD) server. streaming tape buffered read. streaming-tape interface. string. string. string. string operations. string operations. string operations. strings from C programs to implement shared stricat, strcmp, strncmp, strstring, index, rindex: string operations. strip: remove symbols and relocation bits. strincat, strcmp, strncmp, strcpy, strncpy, strlen, strincat, strcmp, strncmp, strcpy, strncpy, strlen, strincat, strcmp, strncmp, strcpy, strncpy, strlen,	ing(3) ing(3) ing(3) ing(3) ose(3S) ose(3S) ose(3S) ic(3S) ic(3S) its(3S) buf(3S) getc(3S) if(1) ror(3S) ind(3) itec(3) ifsend(8) iffer(8) 4) iffer(8) 4) ite(3F) ing(3G) its(3S) itring(3) itr(1) ings(1) itr(1) ings(1) itr(1) ings(1) itr(1) ings(3) itr(1) ings(3) itr(1) itr(1
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a bcopy, bcmp, bzero, ffs: bit and byte strnemp, strepy, strnepy, strlen, index, rindex: extract strings from C programs to implement shared other binary, file. strings: find the printable basename: streat, strneat, stremp, strnemp, strepy, strnepy, index, rindex: string operations. streat, string operations. streat, strneat, stremp,	strcat, strncat, strcmp, strncmp, strncpy, stricmp, strncmp, strcpy, strncpy, strlen, index, strictical strcpy, strncpy, strlen, index, rindex: string strictical stream. stream editor. stream to a remote command. stream to a remote command. stream to Remote Virtual Disk (RVD) server. streaming tape buffered read. streaming-tape interface. string. string. string. string operations. string operations. string operations. string string strictical	ing(3) ing(3) ing(3) ing(3) ose(3S) ose(3S) oben(3S) ek(3S) ic(3S) its(3S) its(3S) buf(3S) getc(3S) if(1) ror(3S) md(3) itec(3) idsend(8) uffer(8) 4) ite(3F) ing(3G) its(3S) itring(3) irr(1) ings(1) irr(1) ings(1) irg(1) ing(3)
rindex: string operations. streat, strneat, operations. streat, strneat, stremp, strnemp, felose, fflush: close or flush a fopen, freopen, fdopen: open a fseek, ftell, rewind: reposition a getchar, fgetc, getw: get character or word from gets, fgets: get a string from a putchar, fputc, putw: put character or word on a puts, fputs: put a string on a setbuffer, setlinebuf: assign buffering to a ungetc: push character back into input sed: ferror, feof, clearerr, fileno: remd, rresvport, ruserok: routines for returning a rexec: return rvdsend - send control tbuffer: st: fdate: return date and time in an ASCII VI_String: draw a gets, fgets: get a puts, fputs: put a bcopy, bcmp, bzero, ffs: bit and byte strnemp, strepy, strnepy, strlen, index, rindex: extract strings from C programs to implement shared other binary, file. strings: find the printable basename: streat, strneat, stremp, strnemp, strepy, strnepy, index, rindex: string operations. streat,	strcat, strncat, strcmp, strncmp, strncpy, strincpy, strncmp, strn	ing(3) ing(3) ing(3) ing(3) ose(3S) ose(3S) ose(3S) ic(3S) ic(3S) its(3S) buf(3S) getc(3S) if(1) ror(3S) ind(3) itec(3) ifsend(8) iffer(8) 4) iffer(8) 4) ite(3F) ing(3G) its(3S) itring(3) itr(1) ings(1) itr(1) ings(1) itr(1) ings(1) itr(1) ings(3) itr(1) ings(3) itr(1) itr(1

	struct: structure Fortran programs	etmost/1\
	struct: structure Fortran programs	struct(1)
struct:		struct(1)
font3812: font		font3812(5)
	stty, gtty: set and get terminal state (defunct)	stty(3C)
	stty: set terminal options	stty(1)
document.	style: analyze surface characteristics of a	style(1)
	su: substitute user id temporarily.	su(1)
alarm: execute a	subroutine after a specified time.	alarm(3F)
VI_Login, VI_Logout: begin logging	subroutine calls and close a log file	log(3G)
VI Init, VI Term: initialize and terminate the	subroutine interface	init(3G)
fetch, store, delete, firstkey, nextkey: data base	subroutines, dbminit,	đbm(3X)
intro: introduction to display graphics	subroutines	intro(3G)
dbm_nextkey, dbm_error, dbm_clearerr: data base	subroutines. /dbm store, dbm delete, dbm firstkey,	ndbm(3)
lib2648:	subroutines for the HP 2648 graphics terminal	lib2648(3X)
su:	substitute user id temporarily.	su(1)
sccs; front end for the SCCS	subsystem.	sccs(1)
sees. From the role are sum:	sum and count blocks in a file.	• •
Suiti.		sum(1)
•	sum: sum and count blocks in a file	sum(1)
du:	· · · · · · · · · · · · · · · · · · ·	du(1)
quot:	summarize file system ownership.	quot(8)
repquota:	summarize quotas for a file system.	repquota(8)
xwininfo - X Window System window information	summarizer	xwininfo(1)
sync: update the	super block	sync(8)
update: periodically update the	super block	update(8)
sync: update	super-block.	sync(2)
suspend: suspend a shell, resuming its	superior.	csh(1)
inetd: internet	"super – server".	inetd(8)
intro: introduction to special files and hardware	support.	intro(4)
		• • •
intro: introduction to special files and hardware	support.	intro(4)
	support: hardware and software support information.	support(1)
support: hardware and software	support information	support(1)
style: analyze	surface characteristics of a document	style(1)
suspend:	suspend a shell, resuming its superior	csh(1)
sleep:	suspend execution for an interval	sleep(1)
sleep:	suspend execution for an interval	sleep(3F)
sleep:	suspend execution for interval	sleep(3)
usleep:	suspend execution for interval	usleep(3)
2000-	suspend: suspend a shell, resuming its superior	csh(1)
interface. ps: Evans and	Sutherland Picture System 2 graphics device	ps(4)
interface. ps. Evans and	swab: swap bytes.	• • •
	swau. swap oytes.	swab(3)
swab:	swap bytes.	swab(3)
swapon: add a	swap device for interleaved paging/swapping	swapon(2)
paging/swapping.	swapon: add a swap device for interleaved	swapon(2)
swapping.	swapon: specify additional device for paging and	swapon(8)
swapon: specify additional device for paging and	swapping	swapon(8)
nextdouble, nextfloat, fptestround, fpsetround,	swapround, fptestflag, fpsetflag, /infinity,	ieee(3)
breaksw: exit from	switch	csh(1)
case: selector in	switch	csh(1)
default: catchall clause in	switch.	csh(1)
endsw: terminate	switch.	csh(1)
	switch: multi-way command branch.	csh(1)
cvtsym: convert	symbol table.	cvtsym(8)
	· · · · · · · · · · · · · · · · · · ·	
makesym: make debugger dbx: dbx	symbol table	makesym(8)
	symbol table information.	dbx(5)
dbx: dbx		dbx(5)
stab:	symbol table types.	stab(5)
readlink: read value of a	symbolic link.	readlink(2)
symlink: make	symbolic link to a file	symlink(2)
strip: remove	symbols and relocation bits	strip(1)
•	symlink: make symbolic link to a file.	symlink(2)
	symorder: rearrange name list	symorder(1)
	sync: update super-block.	sync(2)
	sync: update the super block.	sync(8)
adjtime: correct the time to allow	synchronization of the system clock	adjtime(2)
	synchronize a file's in-core state with that on	fsync(2)
disk. fsync:		7 7 7
select:	synchronous I/O multiplexing	select(2)
csh: a shell (command interpreter) with C-like	syntax	csh(1)
L.	sys: UUCP remote host description file	L.sys(5)
2.1		11/41
2.	syscall: indirect system call	syscall(2)
2	syscall: indirect system call	syscall(8)
perror,	syscall: indirect system call	
	syscall: indirect system call	syscall(8)
perror, terminal.	syscall: indirect system call	syscall(8) perror(3) sysline(1)
perror,	syscall: indirect system call	syscall(8) perror(3) sysline(1) syslog(3)
perror, terminal.	syscall: indirect system call. syscall: system call interface program. sys_errlist, sys_nerr: system error messages. sysline: display system status on status line of a syslog, openlog, closelog, setlogmask: control syslogd: log systems messages.	syscall(8) perror(3) sysline(1) syslog(3) syslogd(8)
perror, terminal.	syscall: indirect system call	syscall(8) perror(3) sysline(1) syslog(3)

psignal,	ave: sigliet, sustant signal proposes	i1/2\
	sys_siglist: system signal messages	psignal(3)
	systat: display system statistics on a crt	systat(1)
interfaces for the IBM Academic Information	Systems experimental display. aedemul: graphics	aedemul(4)
ibmaed, aed: IBM Academic Information	Systems experimental display interface	ibmaed(4)
aedtest: IBM Academic Information	Systems experimental display self-tests	
hy: Network	Systems Hyperchannel interface	
		hy(4)
syslogd: log	systems messages	syslogd(8)
syslogd: log	systems messages	syslogd(8)
kgmon: generate a dump of the operating	system's profile buffers	kgmon(8)
ns: Xerox Network	Systems(tm) protocol family	ns(4F)
rehash: recompute command hash	table	csh(1)
•		
unhash: discard command hash	table	csh(1)
cvtsym: convert symbol	table	cvtsym(8)
keyboard codes: keyboard scancode	table	keyboard codes(5)
makesym: make debugger symbol	table	makesym(8)
		• • •
mkhosts: generate hashed host	table	mkhosts(8)
mkpasswd: generate hashed password	table	mkpasswd(8)
mtab: mounted file system	table	mtab(5)
Remote Virtual Disk (RVD) server configuration	table. rvddb:	rvddb(5)
vwidth: make troff width	table for a font.	vwidth(1)
dbx: dbx symbol	table information	dbx(5)
dbx: dbx symbol	table information	dbx(5)
fdisk: boot record partition	table maintenance utility.	fdisk(8)
getdtablesize: get descriptor	table size	getdtablesize(2)
stab: symbol	table types	stab(5)
htable: convert NIC standard format host	tables.	· · ·
route: manually manipulate the routing		
		, ,
width3812: build width	tables for IBM 3812 Pageprinter fonts	width3812(8)
tbl: format	tables for nroff or troff	tbl(1)
gettable: get NIC format host	tables from a host	gettable(8C)
tabs: set terminal	tabs.	•
taos: set terminai		tabs(1)
	tabs: set terminal tabs	tabs(1)
expand, unexpand: expand	tabs to spaces, and vice versa	expand(1)
ctags: create a	tags file	ctags(1)
otags. ordate a		
	tail: deliver the last part of a file	tail(1)
	tailor: work station customizing assistance	tailor(8)
	talk: talk to another user	talk(1)
talk	talk to another user	talk(1)
tain.		and the same of th
	talkd: remote user communication server	
functions and their inverses. sin, cos,	tan, asin, acos, atan, atan2: trigonometric	sin(3M)
sinh, cosh,	tanh: hyperbolic functions	sinh(3M)
restore Remote Virtual Disk (RVD) packs to and from	tape. savervd, zaprvd, savephys: back up and	savervd(8)
, , , <u>, , , , , , , , , , , , , , , , </u>		
tcopy: copy a mag	tape	tcopy(1)
tm: maninulata	tape archive.	tp(1)
tp: manipulate		
tp. manipulate tar:	tape archive file format.	tar(5)
tar:	•	tar(5)
tar:	tape archiver	tar(1)
tar: tar: tar:	tape archiver	tar(1) tar(1)
tar:	tape archiver	tar(1) tar(1)
tar: tar: tar: tbuffer: streaming	tape archiver	tar(1) tar(1) tbuffer(8)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density	tape archiver	tar(1) tar(1) tbuffer(8) ut(4)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats.	tar(1) tar(1) tbuffer(8) ut(4) tp(5)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen,	tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program.	tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen,	tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program.	tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network.	tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over	tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archive file format.	tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over	tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archive file format. tar: tape archiver.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archive file format. tar: tape archiver.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archive file format. tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archive file format. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archive file format. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore.	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver. tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tbl and eqn constructs.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore.	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver. tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tbl and eqn constructs. tbl: format tables for nroff or troff.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) deroff(1) tbl(1)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore.	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver. tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices. tb! line discipline for digitizing devices. tb! format tables for nroff or troff. tbuffer: streaming tape buffered read.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tbi(1) tbuffer(8)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff,	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver. tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: format tables for nroff or troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tbuffer(8) tc(1)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore.	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver. tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices. tb! line discipline for digitizing devices. tb! format tables for nroff or troff. tbuffer: streaming tape buffered read.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tbl(1) tbuffer(8) tc(1) topen(3F)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff,	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver. tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: format tables for nroff or troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tbl(1) tbuffer(8) tc(1) topen(3F)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff,	tape archiver. tape buffered read. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver. tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: lormat tables for nroff or troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tbuffer(8) tc(1) topen(3F) tcopy(1)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff,	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archive file format. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for for troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tb(1) tbuffer(8) tc(1) topen(3F) tcopy(1) tcp(4P)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff, tape I/O. topen,	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver file format. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: lormat tables for nroff or troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol. tee: pipe fitting.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) deroff(1) tb(4) tb(4) tc(1) topen(3F) tcopy(1) tcp(4P) tee(1)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff, tape I/O. topen,	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: lormat tables for nroff or troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol. tee: pipe fitting. Tektronix 4014.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tb(1) tbuffer(8) tc(1) topen(3F) tcopy(1) tcp(4P)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff, tape I/O. topen,	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver file format. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: lormat tables for nroff or troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol. tee: pipe fitting.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tbl(1) tbuffer(8) tc(1) topen(3F) tcopy(1) tcp(4P) tee(1) tk(1)
tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff, tape I/O. topen, tk: paginator for the last: indicate last logins of users and	tape archiver. tape buffered read. tape buffered read. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for for troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol. tee: pipe fitting. Tektronix 4014. teletypes.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tbl(1) tbuffer(8) tc(1) tcopen(3F) tcopy(1) tcp(4P) tee(1) tk(1) last(1)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff, tape I/O. topen, tk: paginator for the last: indicate last logins of users and index, rindex, lnblnk, len:	tape archiver. tape buffered read. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver. tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for for troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol. tee: pipe fitting. Tektronix 4014. teletypes. tell about character objects.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tbi(1) tbuffer(8) tc(1) topen(3F) tcopy(1) tcp(4P) tee(1) tk(1) last(1) index(3F)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff, tape I/O. topen, tk: paginator for the last: indicate last logins of users and index, rindex, lnblnk, len: log:	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver. tar: tape archiver. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for nroff or troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol. tee: pipe fitting. Tektronix 4014. teletypes. tell about character objects. tell Venus about your password.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) deroff(1) tbuffer(8) tc(1) topen(3F) tcopy(1) tcp(4P) tee(1) tk(1) last(1) index(3F) log(1)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff, tape I/O. topen, tk: paginator for the last: indicate last logins of users and index, rindex, lnblnk, len: log: unlog:	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archive file format. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: lormat tables for nroff or troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol. tee: pipe fitting. Tektronix 4014. teletypes. tell about character objects. tell Venus about your password.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) deroff(1) tb(4) deroff(1) tb(1) topen(3F) tcopy(1) tcp(4P) tee(1) tk(1) last(1) index(3F) log(1) unlog(1)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff, tape I/O. topen, tk: paginator for the last: indicate last logins of users and index, rindex, lnblnk, len: log:	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver file format. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: lormat tables for nroff or troff. touffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol. tee: pipe fitting. Tektronix 4014. teletypes. tell denus about your password. tell Venus about your password. tell Venus you are done with vice. telldir, seekdir, rewinddir, closedir: directory	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) tar(5) tar(1) tar(1) tb(4) deroff(1) tb(4) deroff(1) tbuffer(8) tc(1) topen(3F) tcopy(1) tcp(4P) tee(1) tk(1) last(1) index(3F) log(1) unlog(1) directory(3)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff, tape I/O. topen, tk: paginator for the last: indicate last logins of users and index, rindex, lnblnk, len: log: unlog: operations. opendir, readdir,	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver file format. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: lormat tables for nroff or troff. touffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol. tee: pipe fitting. Tektronix 4014. teletypes. tell denus about your password. tell Venus about your password. tell Venus you are done with vice. telldir, seekdir, rewinddir, closedir: directory	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) tar(5) tar(1) tar(1) tb(4) deroff(1) tb(4) deroff(1) tbuffer(8) tc(1) topen(3F) tcopy(1) tcp(4P) tee(1) tk(1) last(1) index(3F) log(1) unlog(1) directory(3)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff, tape I/O. topen, tk: paginator for the last: indicate last logins of users and index, rindex, lnblnk, len: log: unlog:	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver file format. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: lormat tables for nroff or troff. tbuffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol. tee: pipe fitting. Tektronix 4014. teletypes. tell denus you are done with vice. telldir, seekdir, rewinddir, closedir: directory TELNET protocol server.	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tbl(1) tbuffer(8) tc(1) topen(3F) tcopy(1) tcp(4P) tee(1) tk(1) last(1) index(3F) log(1) unlog(1) directory(3) telnetd(8C)
tar: tar: tar: tar: tbuffer: streaming ut: UNIBUS TU45 tri-density tp: DEC/mag tclose, tread, twrite, trewin, tskipf, tstate: f77 mt: magnetic mt: magnetic mt: magnetic restore.tape, restore.net: install system from network. restore. deroff: remove nroff, troff, tape I/O. topen, tk: paginator for the last: indicate last logins of users and index, rindex, lnblnk, len: log: unlog: operations. opendir, readdir,	tape archiver. tape archiver. tape buffered read. tape drive interface. tape formats. tape I/O. topen, tape manipulating program. tape manipulating program. tape or over network. tape, restore.net: install system from tape or over tar: tape archiver file format. tar: tape archiver. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: line discipline for digitizing devices. tb: lormat tables for nroff or troff. touffer: streaming tape buffered read. tc: photoypesetter simulator. tclose, tread, twrite, trewin, tskipf, tstate: f77 tcopy: copy a mag tape. tcp: Internet Transmission Control Protocol. tee: pipe fitting. Tektronix 4014. teletypes. tell denus about your password. tell Venus about your password. tell Venus you are done with vice. telldir, seekdir, rewinddir, closedir: directory	tar(1) tar(1) tar(1) tbuffer(8) ut(4) tp(5) topen(3F) mt(1) mt(1) restore.tape(8) restore.tape(8) tar(5) tar(1) tar(1) tb(4) tb(4) deroff(1) tbl(1) tbuffer(8) tc(1) topen(3F) tcopy(1) tcp(4P) tee(1) tk(1) last(1) index(3F) log(1) unlog(1) directory(3) telnetd(8C)

su: substitute user id	temporarily	su(1)
signals invalid floating-point operations on a VAX	(temporary). infnan:	infnan(3M)
ptfinstall: install a Program	Temporary Fix (PTF)	ptfinstall(8)
•	termcap: terminal capability data base	termcap(5)
lock: reserve a	terminal	lock(1)
sysline: display system status on status line of a	terminal	sysline(1)
lib2648: subroutines for the HP 2648 graphics	terminal	lib2648(3X)
ttyname, isatty, ttyslot: find name of a	terminal.	ttyname(3)
vhangup: virtually "hangup" the current control	terminal.	vhangup(2)
termcap:	terminal capability data base.	termcap(5)
gettytab:	terminal configuration data base	gettytab(5)
tset:	terminal dependent initialization.	tset(1)
pty: pseudo	terminal driver	pty(4)
xterm: X window system	terminal emulator.	xterm(1)
Xtty: routines to provide	terminal emulator windows	xtty(3X)
tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs:	terminal independent operation routines	termcap(3X)
ttys:	terminal initialization data	ttys(5)
tty: general	terminal interface	tty(4)
tty: general	terminal interface	tty(4)
getty: set	terminal mode	getty(8)
dmf: DMF-32,	terminal multiplexor	dmf(4)
dmz: DMZ-32	terminal multiplexor	dmz(4)
tty: get	terminal name.	tty(1)
, ,		* * . * .
stty: set	terminal options.	stty(1)
ttynam, isatty: find name of a	terminal port.	ttynam(3F)
clear: clear	terminal screen.	clear(1)
script: make typescript of	terminal session.	script(1)
stty, gtty: set and get	terminal state (defunct).	stty(3C)
tabs: set	terminal tabs	tabs(1)
wait: wait for a process to	terminate	wait(3F)
wait, wait3: wait for process to	terminate	wait(2)
exit:	terminate a process.	exit(2)
output. exit:	terminate a process after flushing any pending	exit(3)
kill:	terminate a process with extreme prejudice	kill(1)
abort:	terminate abruptly with memory image	abort(3F)
endif:	terminate conditional	csh(1)
end:	terminate loop.	csh(1)
exit:	terminate process with status	exit(3F)
endsw:	terminate switch.	csh(1)
VI Init, VI Term: initialize and	terminate the subroutine interface	init(3G)
Accelerator, afpacode: load,	test, and bring online the Advanced Floating Point	afpacode(8r)
,	test: condition command	test(1)
drtest; standalone disk	test program.	drtest(8)
sticky: persistent	text and append-only directories.	sticky(8)
ed:	text editor.	ed(1)
ex, edit:		ex(1)
pprint: print	text files on IBM 3812 Pageprinter	pprint(1)
fmt: simple		6
	text formatter	fmt(1)
nroff:	text formatting	nroff(1)
nroff: troff, nroff:		* . * .
	text formatting	nroff(1)
troff, nroff:	text formatting	nroff(1) troff(1)
troff, nroff:	text formatting	nroff(1) troff(1) xtext(3X) tftpd(8C)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines.	text formatting	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum,	text formatting	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent,	text formatting	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag,	text formatting	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr,	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent tgoto, tputs: terminal independent	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin,	nroif(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences;	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction.	nroif(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) diction(1)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile:	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent tgoto, tputs: terminal independent their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle.	nroif(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) tiermcap(3X)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction.	nroif(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) diction(1)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile:	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent tgoto, tputs: terminal independent their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle.	nroif(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle.	nroif(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time. time. time.	nroif(1) troff(1) xtext(3X) tftpd(8C) termcap(3X)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time. time. time.	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified alarm: execute a subroutine after a specified	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time. time. time. time. time. time.	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified alarm: execute a subroutine after a specified etime, dtime: return elapsed execution	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time. time. time. time. time. time. time. time.	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F) etime(3F)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified alarm: execute a subroutine after a specified etime, dtime: return elapsed execution time, ftime: get date and	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time. time. time. time. time. time. time. time. time.	nroff(1) troff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F) etime(3F) time(3C)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified alarm: execute a subroutine after a specified etime, dtime: return elapsed execution time, ftime: get date and time, ctime, ltime, gmtime: return system	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time.	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F) etime(3F) time(3F)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified etime, dtime: return elapsed execution time, ftime: get date and time, ctime, ltime, gmtime: return system ualarm: schedule signal after specified	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time.	nroff(1) troff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F) etime(3F) time(3F) ualarm(3)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified alarm: execute a subroutine after a specified etime, dtime: return elapsed execution time, ftime: get date and time, ctime, ltime, gmtime: return system ualarm: schedule signal after specified time:	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent tgoto, tputs: terminal independent tient inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time.	nroif(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F) etime(3F) time(3F) ualarm(3) time(1)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified etime, dtime: return elapsed execution time, ftime: get date and time, ctime, ltime, gmtime: return system ualarm: schedule signal after specified	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time. time	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F) etime(3F) time(3F) time(3F) ualarm(3) time(1) csh(1)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified alarm: execute a subroutine after a specified etime, dtime: return elapsed execution time, ftime: get date and time, ctime, ltime, gmtime: return system ualarm: schedule signal after specified time:	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time.	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F) etime(3F) time(3F) time(3F) ualarm(3) time(1) csh(1) time(3F)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified alarm: execute a subroutine after a specified etime, dtime: return elapsed execution time, ftime: get date and time, ctime, ltime, gmtime: return system ualarm: schedule signal after specified time:	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time.	nroff(1) troff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F) time(3F) time(3F) time(3F) ualarm(3) time(1) csh(1) time(3F) time(3F)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified alarm: execute a subroutine after a specified etime, dtime: return elapsed execution time, ftime: get date and time, ctime, ltime, gmtime: return system ualarm: schedule signal after specified time:	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time.	nroff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F) etime(3F) time(3F) time(3F) ualarm(3) time(1) csh(1) time(3F)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified etime, dtime: return elapsed execution time, ftime: get date and time, ctime, ltime, gmtime: return system ualarm: schedule signal after specified time: time:	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time.	nroff(1) troff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F) time(3F) time(3F) time(3F) ualarm(3) time(1) csh(1) time(3F) time(3F)
troff, nroff: Xtext: routines to provide simple terminal independent operation routines. independent operation routines. tgetent, tgetnum, independent operation routines. tgetent, operation routines. tgetent, tgetnum, tgetflag, routines. tgetent, tgetnum, tgetflag, tgetstr, acos, atan, atan2: trigonometric functions and diction, explain: print wordy sentences; VI_Tile: at: execute commands at a later gettimeofday, settimeofday: get/set date and shutdown: close down the system at a given alarm: schedule signal after specified alarm: execute a subroutine after a specified etime, dtime: return elapsed execution time, ftime: get date and time, ctime, ltime, gmtime: return system ualarm: schedule signal after specified time: time:	text formatting. text formatting and typesetting. text output windows. tftpd: DARPA Trivial File Transfer Protocol server. tgetent, tgetnum, tgetflag, tgetstr, tgoto, tputs: tgetflag, tgetstr, tgoto, tputs: terminal tgetnum, tgetflag, tgetstr, tgoto, tputs: terminal tgetstr, tgoto, tputs: terminal independent tgoto, tputs: terminal independent operation their inverses. sin, cos, tan, asin, thesaurus for diction. tile a rectangle. time.	nroff(1) troff(1) troff(1) xtext(3X) tftpd(8C) termcap(3X) termcap(3X) termcap(3X) termcap(3X) termcap(3X) sin(3M) diction(1) tile(3G) at(1) gettimeofday(2) shutdown(8) alarm(3C) alarm(3F) time(3F) time(3F) ualarm(3) time(1) csh(1) time(3F) time(3F) time(3F) time(3F)

timed:	time server daemon.	timed(8)
timed.	time time a command.	time(1)
	time: time command	csh(1)
adjtime: correct the	time to allow synchronization of the system clock	adjtime(2)
gmtime, asctime, timezone: convert date and	time to ASCII. ctime, localtime,	ctime(3)
timedo:	timed control program	timedc(8)
	timed: time server daemon	timed(8)
	timedc: timed control program.	timedc(8)
getitimer, setitimer: get/set value of interval	timer	getitimer(2)
times: get process	times	times(3C)
utime: set file	times	utime(3C)
utimes: set file	times	utimes(2)
	times: get process times.	times(3C)
exit, export, login, read, readonly, set, shift,	times, trap, umask, wait: command language. /exec,	sh(1)
ctime, localtime, gmtime, asctime,	timezone: convert date and time to ASCII	ctime(3)
	tk: paginator for the Tektronix 4014	tk(1)
•	tm: TM-11/TE-10 magtape interface	tm(4)
ht:	TM-03/TE-16,TU-45,TU-77 MASSBUS magtape interface.	ht(4)
tm:	TM-11/TE-10 magtape interface.	tm(4)
mt:	TM78/TU-78 MASSBUS magtape interface	mt(4)
4	tmscp: DEC TMSCP magtape interface	tmscp(4)
tmscp: DEC	TMSCP magtape interface	tmscp(4)
	tn3270: full-screen remote login to IBM VM/CMS tn3270: full-screen remote login to IBM VM/CMS	tn3270(1)
isgraph, iscntrl, isascii, toupper, tolower,	tn3270: full-screen remote login to IBM VM/CMS toascii: character classification macros. /isprint,	tn3270(1) ctype(3)
popen, pclose: initiate I/O	to/from a process	popen(3)
lan: IBM RT PC	Token-Ring Adapter	lan(4)
landump: dump IBM	Token-Ring Personal Computer Adapter	landump(8r)
/isprint, isgraph, iscntrl, isascii, toupper,	tolower, toascii: character classification macros	ctype(3)
tstate: f77 tape I/O.	topen, tclose, tread, twrite, trewin, tskipf,	topen(3F)
tsort:	topological sort.	tsort(1)
	touch: update date last modified of a file.	touch(1)
/ispunct, isprint, isgraph, iscntrl, isascii,	toupper, tolower, toascii: character classification/	ctype(3)
, and the second	tp: DEC/mag tape formats	tp(5)
	tp: manipulate tape archive	tp(1)
tgetent, tgetnum, tgetflag, tgetstr, tgoto,	tputs: terminal independent operation routines	termcap(3X)
	tr: translate characters.	tr(1)
ptrace: process	trace	ptrace(2)
ptrace: process	trace	ptrace(2)
trpt: transliterate protocol	trace	trpt(8C)
trsp: transliterate sequenced packet protocol	trace	trsp(8C)
goto: command	transfer	csh(1)
uucico, uucpd:	transfer files queued by uucp or uux	uucico(8C)
ftpd: DARPA Internet File	Transfer Protocol server	ftpd(8C)
tftpd: DARPA Trivial File	Transfer Protocol server	tftpd(8C)
tr:	translate characters	tr(1)
ad: Data	Translation A/D converter	ad(4)
pi: Pascal interpreter code	translator.	pi(1)
trpt:	transliterate protocol trace	trpt(8C)
trsp:	transliterate sequenced packet protocol trace	trsp(8C)
tcp: Internet	Transmission Control Protocol	tcp(4P)
trpfpe, fpecnt:	trap and repair floating point faults	trpfpe(3F) trapov(3F)
trapov: traper:	trap arithmetic errors	traper(3F)
export, login, read, readonly, set, shift, times,	trap, umask, wait: command language. /exec, exit,	sh(1)
porty robing rouse, rouseonly, soc, sinic, tilles,	traper: trap arithmetic errors.	traper(3F)
	trapov: trap and repair floating point overflow	trapov(3F)
I/O. topen, tclose,	tread, twrite, trewin, tskipf, tstate: f77 tape	topen(3F)
topen, tclose, tread, twrite,	trewin, tskipf, tstate: f77 tape I/O	topen(3F)
ut: UNIBUS TU45	tri-density tape drive interface	ut(4)
sin, cos, tan, asin, acos, atan, atan2:	trigonometric functions and their inverses	sin(3M)
tftpd: DARPA	Trivial File Transfer Protocol server	tftpd(8C)
tbl: format tables for nroff or	troff	tbl(1)
Lisp programs to be printed with nroff, vtroff, or	troff. vlp: Format	vlp(1)
ptroff: print	troff files on IBM 3812 Pageprinter	ptroff(1)
	troff, nroff: text formatting and typesetting.	troff(1)
pic:	troff preprocessor for drawing simple pictures	pic(1)
deroff: remove nroff,	troff, tbl and eqn constructs	deroff(1)
vwidth: make faults.	troff width table for a font	vwidth(1)
raults.	trprie, transliterate protocol trace	trpfpe(3F) trpt(8C)
trace.	trsp: transliterate sequenced packet protocol	trsp(8C)
u ace.	true, false: provide truth values.	true(1)
false,	true: provide truth values	false(1)
truncate, ftruncate:	truncate a file to a specified length	truncate(2)
length.	truncate, firuncate: truncate a file to a specified	truncate(2)
iongtii.	, in diseases it allowed a life to a specifical	

false, true: provide	truth values.	false(1)
true, false: provide	truth values.	true(1)
	ts: TS-11 magtape interface	ts(4)
ts:	TS-11 magtape interface	ts(4)
	tset: terminal dependent initialization	tset(1)
topen, tclose, tread, twrite, trewin,	tskipf, tstate: f77 tape I/O	topen(3F)
, , , , , , , , , , , , , , , , , , , ,	tsort: topological sort	tsort(1)
topen, tclose, tread, twrite, trewin, tskipf,	tstate: f77 tape I/O	topen(3F)
topon, tolobo, troud, tillio, trowing takipi,	tty: general terminal interface.	tty(4)
	tty: general terminal interface	* *
		tty(4)
	tty: get terminal name.	tty(1)
	ttynam, isatty: find name of a terminal port	ttynam(3F)
	ttyname, isatty, ttyslot: find name of a terminal	ttyname(3)
getttyent, getttynam, setttyent, endttyent: get	ttys file entry	getttyent(3)
	ttys: terminal initialization data	ttys(5)
ttyname, isatty,	ttyslot: find name of a terminal	ttyname(3)
interface.	tu: VAX-11/730 and VAX-11/750 TU58 console cassette	tu(4)
ut: UNIBUS	TU45 tri-density tape drive interface	ut(4)
		. 1 1
ht: TM-03/TE-16,	TU-45, TU-77 MASSBUS magtape interface	ht(4)
tu: VAX-11/730 and VAX-11/750	TU58 console cassette interface	tu(4)
uu;	TU58/DECtape II UNIBUS cassette interface	uu(4)
ht: TM-03/TE-16,TU-45,	TU-77 MASSBUS magtape interface	ht(4)
tunefs:	tune up an existing file system	tunefs(8)
	tuness: tune up an existing file system	tunefs(8)
topen, tclose, tread,	twrite, trewin, tskipf, tstate: f77 tape I/O	topen(3F)
file: determine file	type	file(1)
	**	2.7
stab: symbol table	types	stab(5)
types: primitive system data	types	types(5)
	types: primitive system data types	types(5)
script: make	typescript of terminal session	script(1)
eqn, neqn, checkeq:	typeset mathematics	eqn(1)
troff, nroff: text formatting and	typesetting	troff(1)
,	ualarm: schedule signal after specified time	ualarm(3)
	uda: UDA-50 disk controller interface	uda(4)
· uda:	UDA-50 disk controller interface	uda(4)
uua.		. 1./
	udp: Internet User Datagram Protocol	udp(4P)
getpw: get name from	uid	getpw(3C)
	ul: do underlining.	ul(1)
	umask: change or display file creation mask	csh(1)
	umask: set file creation mode mask	umask(2)
login, read, readonly, set, shift, times, trap,	umask, wait: command language. /exec, exit, export,	sh(1)
mount,	umount: mount and dismount file system	mount(8)
mount,	umount: mount or remove file system	mount(2)
Ethernet.	un: IBM RT PC Baseband Adapter for use with	un(4)
Bulletineu	unalias: remove aliases.	csh(1)
		• • •
compress,	uncompress, zcat: compress and expand data	compress(1)
ul: do	underlining	ul(1)
xwud - X Window System, window image	undumper	xwud(1)
expand,	unexpand: expand tabs to spaces, and vice versa	expand(1)
•	ungetc: push character back into input stream	ungetc(3S)
	unhash: discard command hash table	csh(1)
uu: TU58/DECtape II	UNIBUS cassette interface.	uu(4)
· · · · · · · · · · · · · · · · · · ·		1.1
up:	unibus storage module controller/drives	up(4)
ut:	UNIBUS TU45 tri-density tape drive interface	ut(4)
	unisdes: remove isdes ed lines.	unifdef(1)
	uniq: report repeated lines in a file	uniq(1)
mktemp: make a	unique file name.	mktemp(3)
gethostid, sethostid: get/set	unique identifier of current host	gethostid(2)
scsiformat: format the IBM 9332 disk	unit.	scsiformat(8c)
flush: flush output to a logical	unit.	flush(3F)
fseek, ftell: reposition a file on a logical	unit.	fseek(3F)
getc, fgetc: get a character from a logical	unit.	getc(3F)
putc, fputc: write a character to a fortran logical	unit.	putc(3F)
dn: DN-11 autocall	unit interface.	dn(4)
	units: conversion program.	units(1)
learn: computer aided instruction about	UNIX	learn(1)
learn: computer aided instruction about	UNIX	learn(1)
reboot;	UNIX bootstrapping procedures	reboot(8)
system: execute a	UNIX command	system(3F)
mtio:	UNIX magtape interface	mtio(4)
muo.		
	unlimit: remove resource limitiations	csh(1)
rmdir, rm: remove	(unlink) directories or files	rmdir(1)
rm, rmdir: remove	(unlink) files or directories.	rm(1)
	unlink: remove a directory entry	unlink(3F)
	unlink: remove directory entry.	unlink(2)
	unlog: tell Venus you are done with vice	unlog(1)
		*

/logb, scalb, rint, classdouble, classfloat, isnan,	unordered, finite, infinity, nextdouble, nextfloat,/	ieee(3)
	unset: discard shell variables	csh(1)
	unsetenv: remove environment variables	
uptime: show how long system has been	up	. ,
tunefs: tune	up an existing file system	`
and from tape. savervd, zaprvd, savephys: back utilities.	up and restore Remote Virtual Disk (RVD) packs to up, down: client Remote Virtual Disk (RVD)	savervd(8) up(1)
vdspin, vdspind: spin	up or spin down a Remote Virtual Disk (RVD)	
vaspin, vaspina spin	up: unibus storage module controller/drives	up(4)
touch:	update date last modified of a file	- 1 1
	update: periodically update the super block	update(8)
sync:	update super-block	sync(2)
sync:	update the super block	sync(8)
update: periodically	update the super block	
spinup, spindown: spin	up/down Remote Virtual Disk (RVD) pack	
du: summarize disk	uptime: show how long system has been up	
quota: display disc	usage and limits.	du(1) quota(1)
un: IBM RT PC Baseband Adapter for	use with Ethernet.	
cvt00to12: convert IBM 3820 and IBM 3800 fonts for	use with the IBM 3812 Pageprinter. /cvt20to12,	
what: show what versions of object modules were	used to construct a file	what(1)
login: login new	user	csh(1)
talk: talk to another	user	talk(1)
write: write to another	user	write(1)
seteuid, setruid, setgid, setegid, setrgid: set	user and group ID. setuid,	setuid(3)
talkd: remote	user communication server	talkd(8C)
udp: Internet	User Datagram Protocol	udp(4P)
whoami: print effective current su: substitute	user id temporarily.	whoami(1) su(1)
getuid, geteuid: get	user identity.	getuid(2)
setreuid: set real and effective	user ID's.	setreuid(2)
finger:	user information lookup program	finger(1)
fingerd: remote	user information server	fingerd(8C)
nfcomment: a	user interface to the notesfile system	nfcomment(3)
whois: DARPA Internet	user name directory service	whois(1)
getuid, getgid: get	user or group ID of the caller	getuid(3F)
edquota: edit	user quotas	
xset - X window system getusershell, setusershell, endusershell: get legal	user shells	xset(1) getusershell(3)
getusershen, setusershen, endusershen. get legar	USERFILE: UUCP pathname permissions file	USERFILE(5)
adduser: procedure for adding new	users.	adduser(8)
binmail: send or receive mail among	users.	binmail(1)
wall: write to all	users	wall(1)
last: indicate last logins of	users and teletypes.	last(1)
	users: compact list of users who are on the system.	users(1)
getlog: get	user's login name.	getlog(3F)
users: compact list of (SCSI) Adapter. sc: IBM 9332 disks	users who are on the system	
(SCS1) Adapter. sc: 1BM 9332 disks	usleep: suspend execution for interval	
	ut: UNIBUS TU45 tri-density tape drive interface	
up, down: client Remote Virtual Disk (RVD)	utilities.	up(1)
xmodmap, xprkbd - X Window System keyboard modifier	utilities	xmodmap(1)
fdisk: boot record partition table maintenance	utility.	
minidisk: minidisk maintenance	utility.	
xrdb - Server Resource Database	Utility	
X window system root window parameter setting	utility xsetroot:	
consoles: sautil: standalone	utility database of display screens	
getrusage: get information about resource	utilization.	• • • • • • • • • • • • • • • • • • • •
vtimes: get information about resource	utilization.	
Ü	utime: set file times.	
	utimes: set file times	
	utmp, wtmp: login records.	
	uu: TU58/DECtape II UNIBUS cassette interface	
uux.	uucico, uucpd: transfer files queued by uucp or	
rmail: handle remote mail received via	uuclean: uucp spool directory clean-up	uuclean(8C)
rmail: nandle remote mail received via L-devices:	UUCP device description file	
uuxqt:	UUCP execution file interpreter	
L.aliases:	UUCP hostname alias file.	
uucico, uucpd: transfer files queued by	uucp or uux.	
USERFILE:	UUCP pathname permissions file	USERFILE(5)
L-dialcodes:	UUCP phone number index file	
L.cmds:	UUCP remote command permissions file	
L.sys:	UUCP remote host description file	
uupoll: poll a remote	UUCP site	uupoli(8C)

	man and the state of the state	
uuclean:	uucp spool directory clean-up	` '
uusnap: show snapshot of the	UUCP system.	uusnap(8C)
uucico,	uucpd: transfer files queued by uucp or uux	
uuencode: format of an encoded	uuencode file.	
	uuencode: format of an encoded uuencode file	uuencode(5)
	uupoll: poll a remote UUCP site	
	uusnap: show snapshot of the UUCP system	uusnap(8C)
uucico, uucpd: transfer files queued by uucp or	uux	
	uuxqt: UUCP execution file interpreter	uuxqt(8C)
.PP	uwm - Window Manager Client Application of X .PP.	uwm(1)
	va: Benson-Varian interface.	
vacation: return "I am on	vacation" indication.	
vacation, leturn 1 am on	vacation indication	
	vacation: return "I am on vacation" indication.	` '
	valloc: aligned memory allocator.	
abs: integer absolute	value	abs(3)
hypot, cabs: Euclidean distance, complex absolute	value	hypot(3M)
fabs, floor, ceil: absolute	value, floor, ceiling functions	floor(3M)
getenv:	value for environment name	getenv(3)
readlink: read	value of a symbolic link	readlink(2)
getenv: get	value of environment variables.	getenv(3F)
getitimer, setitimer: get/set	value of interval timer.	
set: change	value of shell variable.	csh(1)
false, true: provide truth	values.	
true, false: provide truth	values.	` '
dfimin, dfimax, dffrac, inmax: return extreme	values. filmin, filmax, ffrac,	, , ,
rand, drand, irand: return random	values.	
htoni, htons, ntohl, ntohs: convert	values between host and network byte order	
	varargs: variable argument list	varargs(3)
set: change value of shell	variable	
varargs:	variable argument list	
seteny: set	variable in environment.	
@: arithmetic on shell		` '
		csh(1)
unset: discard shell	variables	
unsetenv: remove environment	variables	csh(1)
getenv: get value of environment	variables	getenv(3F)
crl:	VAX 8600 console RL02 interface	cri(4)
signals invalid floating-point operations on a	VAX (temporary). infnan:	infnan(3M)
as:	VAX-11 assembler	
cons;	VAX-11 console interface	` '
interface. tu:	VAX-11/730 and VAX-11/750 TU58 console cassette .	tu(4)
		, ,
tu: VAX-11/730 and	VAX-11/750 TU58 console cassette interface	tu(4)
	vdabort: abort and spin down a drive	vdabort(8)
	vddb: Remote Virtual Disk (RVD) data base manager	vddb(8)
Virtual Disk (RVD).	vdspin, vdspind: spin up or spin down a Remote	vdspin(2)
(RVD). vdspin,	vdspind: spin up or spin down a Remote Virtual Disk	vdspin(2)
statistics.	vdstats: acquire client Remote Virtual Disk (RVD)	vdstats(2)
statistics.	vdstats: list client Remote Virtual Disk (RVD)	vdstats(8)
log: tell	Venus about your password	
unlog: tell	Venus you are done with vice	U ,
		• • • • • • • • • • • • • • • • • • • •
assert: program	verification.	assert(3)
lint: a C program	verifier	lint(1)
expand, unexpand: expand tabs to spaces, and vice	versa.	expand(1)
vfont: font formats for the Benson-Varian or	Versatec	vfont(5)
vp:	Versatec interface	vp(4)
file. what: show what	versions of object modules were used to construct a .	what(1)
Versatec.	viont: font formats for the Benson-Varian or	vfont(5)
efficient way.	vfork: spawn new process in a virtual memory	vfork(2)
	vgrind: grind nice listings of programs	vgrind(1)
3812 Pageprinter.	vgrind: grind nice listings of programs for the IBM	vgrind(1)
Sold I agoptiment	vgrindefs: vgrind's language definition data base	vgrindefs(5)
varindefe:		
vgrindefs:	vgrind's language definition data base	vgrindefs(5)
terminal.	vhangup: virtually "hangup" the current control	vhangup(2)
on ex.	vi: screen oriented (visual) display editor based	vi(1)
rmail: handle remote mail received	via uucp.	
	VI_ALine, VI_RLine: draw a line	line(3G)
	VI_AMove, VI_RMove: move the current point	move(3G)
unlog: tell Venus you are done with	vice	unlog(1)
expand, unexpand: expand tabs to spaces, and	vice versa.	expand(1)
	VI Circle: draw a circle.	circle(3G)
	VI Clip: set clipping window	clip(3G)
	VI_Color: change screen color.	color(3G)
	VI_Copy: copy an area	copy(3G)
And the second of		dash(3G)
VI MD-f-C VI ED-f-C VI E-C	VI_Dash: set line dash pattern	
VI_MDefnCur, VI_FDefnCur, VI_EnCur,	VI DisCur, VI PosnCur: control the display cursor	cursor(3G)
	VI DisCur, VI PosnCur: control the display cursor	cursor(3G)
VI_MDefricur, VI_FDefricur, VI_Encur, VI_Font, VI_GetFont, display cursor. VI_MDefricur, VI_FDefricur,	VI_DisCur, VI_PosnCur: control the display cursor	

more, page: file perusal filter for crt	viewing.	more(1)
control the display cursor. VI_MDefnCur, VI_MImage,	VI_FDefnCur, VI_EnCur, VI_DisCur, VI_PosnCur: . VI_FImage: draw an image	cursor(3G) image(3G)
manipulate fonts.	VI_Font, VI_GetFont, VI_DropFont: select and	font(3G)
•	VI_Force: force output of graphics orders	force(3G)
VI_MRead,	VI_FRead: read display data	read(3G)
fonts. $\overline{V}I$ —Font, subroutine interface.	VI_GetFont, VI_DropFont: select and manipulate VI_Init, VI_Term: initialize and terminate the	font(3G) init(3G)
and close a log file.	VI_Login, VI_Logout: begin logging subroutine calls	log(3G)
a log file. VI_Login,	VI_Logout: begin logging subroutine calls and close .	log(3G)
VI_PosnCur: control the display cursor.	VI_MDefnCur, VI_FDefnCur, VI_EnCur, VI_DisCur, VI_Merge: set merge mode	cursor(3G) merge(3G)
	VI MImage, VI FImage: draw an image	image(3G)
	VI_MRead, VI_FRead: read display data	read(3G)
VI_MDefnCur, VI_FDefnCur, VI_EnCur, VI_DisCur,	VI PosnCur: control the display cursor	cursor(3G)
VI QPoint, VI QWidth: query graphics parameters.	vipw: edit the password file	vipw(8) , query(3G)
VI_QPoint, VI_QWidth: query graphics/ VI_QClip,	VI_QColor, VI_QDash, VI_QFont, VI_QMerge,	query(3G)
VI_QWidth: query graphics/ VI_QClip, VI_QColor,	VI_QDash, VI_QFont, VI_QMerge, VI_QPoint,	query(3G)
graphics/ VI_QClip, VI_QColor, VI_QDash,	VI_QFont, VI_QMerge, VI_QPoint, VI_QWidth: query	query(3G)
VI_QClip, VI_QColor, VI_QDash, VI_QFont, VI_QClip, VI_QColor, VI_QDash, VI_QFont, VI_QMerge,	VI_QMerge, VI_QPoint, VI_QWidth: query graphics/ VI_QPoint, VI_QWidth: query graphics parameters	query(3G) query(3G)
VI_QDash, VI_QFont, VI_QMerge, VI_QPoint,	VI_QWidth: query graphics parameters. /VI_QColor, .	query(3G)
VI_ALine,	VI_RLine: draw a line	line(3G)
VI_AMove, rvd: Remote	VI_RMove: move the current point	move(3G)
newvd: create a new filesystem on a Remote	Virtual Disk protocol	rvd(4p) newvd(8)
vdspin, vdspind: spin up or spin down a Remote	Virtual Disk (RVD)	vdspin(2)
vddb: Remote	Virtual Disk (RVD) data base manager	vddb(8)
rvddown: force spindown of a Remote spinup, spindown: spin up/down Remote	Virtual Disk (RVD) pack	rvddown(8) spinup(8)
rvdexch: exchange names of two Remote	Virtual Disk (RVD) packs	rvdexch(8)
rvdflush: spindown client's Remote	Virtual Disk (RVD) packs.	rvdflush(8)
/zaprvd, savephys: back up and restore Remote	Virtual Disk (RVD) packs to and from tape	savervd(8)
rvdchlog: change logging level of Remote rvdgetm: get operations message from Remote	Virtual Disk (RVD) server	rvdchlog(8) rvdgetm(8)
rvdsend - send control stream to Remote	Virtual Disk (RVD) server	rvdsend(8)
rvdsetm: set operations message on Remote	Virtual Disk (RVD) server	rvdsetm(8)
rvdshow: show connections to Remote rvdshut: force shutdown of Remote	Virtual Disk (RVD) server	rvdshow(8) rvdshut(8)
ryddb: Remote	Virtual Disk (RVD) server	rvddb(5)
rvdsrv: Remote	Virtual Disk (RVD) server daemon	rvdsrv(8)
rvdlog: cause Remote	Virtual Disk (RVD) server to log statistics	rvdlog(8)
vdstats: acquire client Remote vdstats: list client Remote	Virtual Disk (RVD) statistics	vdstats(2) vdstats(8)
up, down: client Remote	Virtual Disk (RVD) utilities	up(1)
/etc/rvd/rvdtab: information about client Remote	Virtual Disks (RVDs)	rvdtab(5)
vfork: spawn new process in a	virtual memory efficient way.	vfork(2)
vmstat: report vmstat: report	virtual memory statistics	vmstat(1) vmstat(1)
vhangup:	virtually "hangup" the current control terminal	vhangup(2)
	VI_Run: process a log file	run(3G)
vi: screen oriented	VI_String: draw a string	string(3G) vi(1)
interface. VI Init,	VI Term: initialize and terminate the subroutine	init(3G)
· · ·	VI_Tile: tile a rectangle	tile(3G)
	VI Width: set line width	width(3G)
consumption. vtroff, or troff.	vlp: Format Lisp programs to be printed with nroff,	vlimit(3C) vlp(1)
tn3270: full-screen remote login to IBM	VM/CMS	tn3270(1)
tn3270: full-screen remote login to IBM	VM/CMS	tn3270(1)
	vmstat: report virtual memory statistics	vmstat(1)
fs, inode: format of file system	volume	vmstat(1) fs(5)
	vp: Versatec interface	vp(4)
vint Cormet Lies announced to be united with the	vtimes: get information about resource utilization	vtimes(3C)
vlp: Format Lisp programs to be printed with nroff,	vtroff, or troff	vlp(1) vv(4)
	vwidth: make troff width table for a font.	vwidth(1)
	w: who is on and what they are doing	w(1)
road roadonly sat shift times two	wait: await completion of process	wait(1)
read, readonly, set, shift, times, trap, umask, wait:	wait: command language. /exec, exit, export, login, wait for a process to terminate	sh(1) wait(3F)
wait:	wait for background processes to complete	csh(1)
sigpause: atomically release blocked signals and	wait for interrupt.	sigpause(2)
wait, wait3:	wait for process to terminate	wait(2)

	and the same trace and a same trace as	** (2.57)
	wait: wait for a process to terminate	wait(3F)
	wait: wait for background processes to complete	csh(1)
	wait, wait3: wait for process to terminate	wait(2)
wait,	wait3: wait for process to terminate	wait(2)
atq: print the queue of jobs	waiting to be run.	atq(1)
	wall: write to all users	wall(1)
	wc: word count	wc(1)
what: show what versions of object modules	were used to construct a file	what(1)
whatis: describe	what a command is.	whatis(1)
crash:	what happens when the system crashes	crash(8r)
crash:	what happens when the system crashes	crash(8V)
used to construct a file.	what: show what versions of object modules were	what(1)
w: who is on and	what they are doing.	w(1)
construct a file. what: show	what versions of object modules were used to	what(1)
	whatis: describe what a command is	whatis(1)
crash: what happens	when the system crashes	crash(8r)
crash: what happens	when the system crashes	crash(8V)
leave: remind you	when you have to leave	leave(1)
program.	whereis: locate source, binary, and or manual for	whereis(1)
paths (csh only).	which: locate a program file including aliases and	which(1)
exec, exit, export, login,/ sh, for, case, if,	while, :, ., break, continue, cd, eval,	sh(1)
exec, exit, export, logili, sii, loi, ease, ii,	4.1	
1		csh(1)
break: exit	while/foreach loop	csh(1)
users: compact list of users	who are on the system.	users(1)
from:	who is my mail from?	from(1)
w:	who is on and what they are doing	w(1)
who:	who is on the system	who(1)
biff: be notified if mail arrives and	who it is from.	biff(Ì)
,	who: who is on the system.	who(1)
	who ami: print effective current user id	whoami(1)
	whois: DARPA Internet user name directory service	
VI Width, and line	·	whois(1)
VI_Width: set line	width.	width(3G)
fold: fold long lines for finite	width output device	fold(1)
vwidth: make troff	width table for a font.	vwidth(1)
width3812: build	width tables for IBM 3812 Pageprinter fonts	width3812(8)
Pageprinter fonts.	width3812: build width tables for IBM 3812	width3812(8)
VI Clip: set clipping	window.	clip(3G)
xpr: print X	window dump.	xpr(1)
window:	window environment.	window(1)
xwd - X Window System,	window image dumper	xwd(1)
· · · · · · · · · · · · · · · · · · ·	,	**
xwud - X Window System,	window image undumper	xwud(1)
xwininfo - X Window System	window information summarizer	xwininfo(1)
xmh: X	window interface to the mh Mail Handler	xmh(1)
wm: a simple real-estate-driven	window manager	wm(1)
.PP uwm -	Window Manager Client Application of X .PP	uwm(1)
xsetroot: X window system root	window parameter setting utility	xsetroot(1)
bitmap: bitmap editor for X	window system	bitmap(1)
xhost - X	window system access control program	xhost(1)
xclock - X	Window System, analog / digital clock	xclock(1)
xfd - X	window system font displayer	xfd(1)
xlsfonts - X		xlsfonts(1)
xinit - X	window system initializer	xinit(1)
Xlib: C Language X	Window System Interface Library	Xlib(3X)
xmodmap, xprkbd - X	Window System keyboard modifier utilities	xmodmap(1)
xload - X	window system load average display	xload(1)
xprop - X	Window System property displayer	xprop(1)
utility. xsetroot: X	window system root window parameter setting	xsetroot(1)
xterm: X	window system terminal emulator	xterm(1)
xset - X	window system user setup program	xset(1)
xwd - X	Window System, window image dumper	` '
		xwd(1)
xwud - X	Window System, window image undumper	xwud(1)
xwininfo - X	Window System window information summarizer	xwininfo(1)
•	window: window environment.	window(1)
Xtext: routines to provide simple text output	windows	xtext(3X)
Xtty: routines to provide terminal emulator	windows	xtty(3X)
xrefresh - refresh all	windows on the screen	xrefresh(1)
fastboot, fasthalt: reboot/halt the system	without checking the disks	fastboot(8)
in the system is a system of the system	wm: a simple real-estate-driven window manager	wm(1)
WC:	word count.	wc(1)
getc, getchar, fgetc, getw: get character or	word from stream	getc(3S)
putc, putchar, fputc, putw: put character or	word on a stream.	putc(3S)
diction, explain: print	wordy sentences; thesaurus for diction	diction(1)
tailor:	work station customizing assistance	tailor(8)
cd: change	6	
	working directory	cd(1)
chdir: change current	working directory	cd(1) chdir(2)

pwd:	working directory name	pwd(1)
getwd: get current	working directory pathname	getwd(3)
putc, fputc: dosread: read.	write, dir, delete on PC-DOS diskette.	putc(3F) dosread(1)
write, writev:	write output.	write(2)
wall:	write to all users.	wall(1)
write:	write to another user.	write(1)
	write: write to another user.	write(1)
	write, writev: write output.	write(2)
write,	writev: write output.	write(2)
open: open a file for reading or	writing, or create a new file	open(2)
utmp,	wtmp: login records.	utmp(5)
xcalc:	X based scientific calculator	xcalc(1)
XMenu -	X Deck of cards Menu System	XMenu(3X)
events. xemul:	X input emulator for queuing keyboard and mouse	xemul(4)
.PP uwm - Window Manager Client Application of		uwm(1)
xpr: print	X window dump.	xpr(1)
xmh:	X window interface to the mh Mail Handler	xmh(1)
bitmap: bitmap editor for xhost -	X window system	bitmap(1) xhost(1)
xclock -	X Window System, analog / digital clock	xclock(1)
xfd -		xfd(1)
xisfonts -		xlsfonts(1)
xinit -	X window system initializer	xinit(1)
Xlib: C Language	X Window System Interface Library	Xlib(3X)
xmodmap, xprkbď -	X Window System keyboard modifier utilities	xmodmap(1)
xload -	X window system load average display	xload(1)
xprop -	X Window System property displayer	xprop(1)
utility. xsetroot:	X window system root window parameter setting	xsetroot(1)
xterm:	X window system terminal emulator	xterm(1)
xset -	X window system user setup program	xset(1)
- bwx - wud -	X Window System, window image dumper X Window System, window image undumper	xwd(1)
xwdd - xwininfo -	X Window System, window information summarizer	xwud(1) xwininfo(1)
ddn: DDN Standard Mode	X.25 IMP interface	ddn(4)
ddii. 19514 Staildaid Mode	xcalc: X based scientific calculator.	xcalc(1)
•	xclock - X Window System, analog / digital clock	xclock(1)
mouse events.	xemul: X input emulator for queuing keyboard and	xemul(4)
en:	and the same of th	en(4)
idp:	Xerox Internet Datagram Protocol	idp(4P)
ns:	Xerox Network Systems(tm) protocol family	ns(4F)
ns_addr, ns_ntoa:	Xerox NS(tm) address conversion routines	ns(3N)
spp:	Xerox Sequenced Packet Protocol	spp(4P)
	xfd - X window system font displayer	xfd(1)
xsend,	0 ,	xsend(1)
	xhost - X window system access control program	xhost(1)
	xinit - X window system initializer	xinit(1)
	Xlib: C Language X Window System Interface Library.	Xlib(3X)
	xload - X window system load average display xlsfonts - X window system font list displayer	xload(1) xlsfonts(1)
	XMenu - X Deck of cards Menu System	XMenu(3X)
utilities.	xmodmap, xprkbd - X Window System keyboard modifier	xmodmap(1)
diffico.	XNSrouted: NS Routing Information Protocol daemon.	XNSrouted(8C)
bit: and, or,	xor, not, rshift, Ishift bitwise functions	bit(3F)
5.5. mig, 61,	xpr: print X window dump	xpr(1)
utilities. xmodmap,	xprkbd - X Window System keyboard modifier	xmodmap(1)
range di santa di sa	xprop - X Window System property displayer	xprop(1)
	xrdb - Server Resource Database Utility	xrdb(1)
	xrefresh - refresh all windows on the screen	xrefresh(1)
	xsend, xget, enroll: secret mail.	xsend(1)
	xset - X window system user setup program	xset(1)
setting utility. shared strings.	xsetroot: X window system root window parameter xstr: extract strings from C programs to implement	xsetroot(1) xstr(1)
snated strings.	xstr: extract strings from C programs to implement	xterm(1)
windows.	Xtext: routines to provide simple text output	xtext(3X)
windows.	Xtty: routines to provide terminal emulator	xtty(3X)
7	xwd - X Window System, window image dumper	xwd(1)
summarizer	xwininfo - X Window System window information	xwininfo(1)
	xwud - X Window System, window image undumper	xwud(1)
j0, j1, jn,		j0(3M)
j0, j1, jn, y0,		j0(3M)
	yacc: yet another compiler-compiler	yacc(1)
	yes: be repetitively affirmative	yes(1)
j0, j1, jn, y0, y1,		j0(3M)
Virtual Disk (RVD) packs to and from/ savervd,	zaprvd, savephys: back up and restore Remote	savervd(8)
compress, uncompress,	zcat: compress and expand data	compress(1)