P-H Kamp

UNIX™

A Quick Reference Guide to Zilog's Enhanced Unix System

Systems
Zilog
an affiliate of EXXON Corporation

03-3269-01

March 1984

INTRODUCTION

A QUICK REFERENCE GUIDE TO ZILOG'S ENHANCED UNIX SYSTEM summarizes many popular features of the UNIX operating system. All entries are brief, and are intended to serve as a memory aid for the experienced UNIX user. More explanation can be found in the appropriate System 8000 manual. The complete set contains the:

System 8000 ZEUS Administrator Manual	03-3246
System 8000 ZEUS Languages/ Programming Tools Manual	03-3249
System 8000 ZEUS Utilities Manual	03-3250
System 8000 ZEUS Reference Manual	03-3255

Throughout this guide, two conventions are used to describe command and routine formats. Text appearing in **bold** type is entered literally, and text appearing in *italics* is replaced by the user with a specified value.

TABLE OF CONTENTS

USER COMMANDS

The following list of commands, taken from Section 1 of the ZEUS Reference Manual, includes a name and description line, command synopsis line, and a short explanation of the options and variables. The commands are available to all users and are arranged alphabetically.

To execute a command, type all boldface text literally. Options are enclosed in brackets ([]) and should be implemented where necessary. Substitute actual values for the italicized portion of the command line. Ellipses (...) denote the preceding parameter can be repeated as needed.

adb is a general purpose debugger; it examines files and provides a controlled environment for execution of ZEUS programs.

adb [**−w**] [*objfil* [*corfil*]]

−w	creates and opens *objfil* and *corfil* for reading, modifying and writing.
objfil	is an executable program file, preferably containing a symbol table; default is a.out.
corfil	is a core image file produced after executing *objfil*, default is core.

admin creates and administers Source Code Control System (SCCS) files and changes parameters of existing ones.

admin *[options] files*

−a*login*	adds a *login* name, or numerical ZEUS group ID, to the list of users who may make deltas (changes) to the SCCS file.
−d*flag*	removes the specified *flag* from an SCCS file.
−e*login*	erases a *login* name, or numerical group ID, from the list of users who may make deltas (changes) to the SCCS file.
−f*flag*	places a *flag* and its possible value in the SCCS file.
−h	checks the SCCS file structure and compares a new checksum with the stored checksum.
−i[*name*]	*name* is the file from which text for a new SCCS file is taken.
−m[*mrlist*]	inserts modification request (*MR*) numbers into the SCCS file as the reason for creating the initial delta in a manner identical to **delta**.
−n	indicates creation of a new SCCS file.
−r*rel*	names the release to insert into an initial delta.

–t[*name*]	*name* is a file from which descriptive text is taken.
–y[*comment*]	inserts *comment* text into the SCCS file describing the reason for making the delta.
–z	recomputes and stores first line checksums.

apropos locates manual entries by keyword lookup.

apropos *word* ...

ar maintains groups of files combined into an archive file.

ar [**drqtpmx**] [**vualbcl**] *afile files* ...

d	deletes files from archive.
r	replaces files in archive; with **u**, replaces files dated later than archive date; can be placed after (**a**) or before (**b** or **i**) *posname*; otherwise it is appended.
q	quickly appends files to archive.
t	prints archive's table of contents.
p	prints named files in archive.
m	moves files to end of archive; with **a**, **b** or **i**, *posname* must be present.
x	extracts files.
c	creates *afile*.
l	places temporary files in the local directory instead of in /tmp.
v	gives a file-by-file description of the creation of a new archive file. With **t**, gives a long listing of all information about the files. With **p**, precedes each file with a name.
afile	archive filename.

as is the PLZ/ASM assembler; it assembles the named file.

as [**–flopuz**] *file*

–f	assembles floating point instructions.
–l	produces a listing of object code and locations in *file*.l.
–o *objfile*	leaves output of assembly in *objfile*; default is in the file a.out.
–p	writes a listing to standard output.
–u	treats undefined references as externals.
–z	produces Zobj object format for MCZ compatible systems. The default output file is t.out instead of a.out.

at executes commands in a given file at a specified time and day.

at *time* [*day*] *file*

awk scans input files for patterns specified in a program file.

awk [**–Fc**] [*progfile*] [*file*] ...
or
awk [**–Fc**] [**–f** *progfile*] [*file*] ...

–Fc	use c as field separator.
–f	use next argument as *progfile*.

banner prints arguments (up to 10 characters each) in large letters on the standard output.

banner *strings*

basename deletes prefixes ending in "/" and suffixes (if present) from a string, and prints to the standard output.

basename *string* [*suffix*]

bc interactively translates a language resembling C and provides unlimited precision arithmetic.

bc [**–cl**] [*file* ...]

–c	compiles, but does not run **dc**; the **dc** input appears on **bc's** standard output.
–l	defines a math function library.

bdiff finds lines which must change for two large files to agree.

bdiff *file1 file2* [*n*] [**–s**]

n	line segment number (3500 by default) for comparing files.
–s	suppresses diagnostics printed by **bdiff.**

cal prints a calendar for the specified year and/or month.

cal [*month*] *year*

month	decimal number from 1 to 12.
year	decimal number from 1 to 9999.

calendar prints daily reminders from a calendar file in the current directory.

calendar [–]

–	sends reminders via mail to every user having a calendar file in their login directory.

cas is the ZEUS assembler.

cas [**–dlou**] *file*

–d	includes internal labels in the a.out symbol table.
–l	produces a listing of object code and addresses in *file*.l.
–o *objfile*	leaves output of assembly on *objfile*; default is to the file a.out.
–u	treats all undefined references as externals.

cat concatenates and prints files.

cat [**−s**] [**−u**] *file* ...

−	reads from the standard output; same as if not giving an output file.
−s	makes **cat** silent about non-existent files.
−u	output is unbuffered.

cb reformats a C source file, providing spacing and indentation to improve readability of the listing.

cb < *file*.**c**

cc is the portable C compiler modified to create Z8000 code.

cc [*option*] *file*

−c	suppresses loading.
−D*name*	defines *name* to equal 1 to the preprocessor.
−D*name*=*def*	defines *name* to the preprocessor.
−E	runs only the macro preprocessor and sends the result to the standard output.
−I*dir*	brings in a *dir*ectory of #include files.
−Ol	invokes the C global optimizer to apply loop optimization.
−Or	invokes the C global optimizer to apply loop optimization and register allocation.
−O	invokes the C peephole optimizer for Z8000 code.
−p	produces code to be used by **prof**(1).
−P	preprocesses only; output to *file*.i.
−S[l]	compiles, but suppresses assembly and linking steps. With **l** source lines are used as assembly language comments.
−U*name*	removes any initial definition of *name*.

cdc changes the delta commentary, for the SID specified by the −r keyletter, of each named SCCS file.

cdc **−r***SID* [**−m**[*mrlist*]] [**−y**[*comment*]] *files*

−m[*mrlist*]	with the **v** flag set in the SCCS file, the list of *MR* numbers is added and/or deleted in the delta commentary of the *SID* specified by the **−r** keyletter.
−r*SID*	specifies the (*SID*) string for the delta commentary to be changed.
−y[*comment*]	replaces the existing *comment*(s) for the delta specified by the **−r** keyletter.

chgrp changes the group-ID of files.

chgrp *group file* ...

checkcw (see **cw** entry) checks that left and right delimiters and .CW/.CN pairs balance. Prints offending lines.

checkcw [**−l***xx*] [**−r***xx*] *files*

checkeq (see **eqn** entry) reports missing or unbalanced delimiters and .EQ/.EN pairs.

checkeq [*file*] ...

chkdiff lists differences between versions of a file under Zilog Source Control.

chkdiff [**−h**] [**−v** *rel.lev*] [**−v** *rel.lev*] *file*

−h	invokes the "halfhearted" version of **diff**(1).
−v	lists differences between the source file and the specified version. If used twice, lists the differences between two specified versions.

chkin checks in a source file to its Zilog Source Control file.

chkin [**−b**] [**−c** *comment*] [**−d** *dir*] [**−r**] *file* ...

−b	bumps the release number.
−c *comment*	inserts *comment* as a comment line enclosed in double quotes.
−d *dir*	gets the source from directory *dir* instead of the working directory.
−r	removes rather than replaces the source file with a read-only file.

chkout reconstructs any version of a source file under Zilog Source Control.

chkout [**−d** *dir*] [**−e**] [**−h**] [**−p**] [**−v** *rel.lev*] *file* ...

−d *dir*	creates the source file in the directory *dir* instead of the working directory.
−e	checks out the version as an editable file.
−h	lists history of the control file.
−p	lists the version on the standard output. Substitutes keywords.
−v *rel.lev*	checks out the specified version instead of the last version.

chkwhat prints Zilog Source Control what strings contained in specified file.

chkwhat [**−w**] *file* ...

−w	prints the entire what string.

chmod changes the permission mode of designated files and directories.

chmod *mode file* ...

Mode:	*Bits*:	*Meaning*:
0	---	no permissions
1	--x	execute (search in directory) only
2	-w-	write only
3	-wx	write and execute (search)
4	r--	read only
5	r-x	read and execute (search)
6	rw-	read and write
7	rwx	read, write and execute (search)

chown changes the owner of files.

chown *owner file*

cmp compares two files.

cmp [**−l**] [**−s**] *file1 file2*

−l	prints the byte number (decimal) and the differing bytes (octal) for each difference.
−s	print nothing for differing files; return codes only.

code prints characters with their hex equivalents.

code [< *file*]

col is an **nroff** post-processing filter that strips out escape sequences for printer output.

col [**−bfx**] [< *file*]

−b	generates output suitable for a device that cannot backspace.
−f	eliminates all reverse motion but permits halfline-forward (ESC-9) sequences.
−x	does not generate new tab characters.

comb generates a shell procedure which reconstructs the given SCCS files.

comb [**−c***list* **−o** **−p***SID* **−s**] *files*

−c*list*	is a *list* of deltas to be preserved; discards all others.
−o	for each **get** **−e** generated, accesses the reconstructed file at the release of the created delta; otherwise, the most recent reconstructed file is accessed.
−p*SID*	specifies the *SID* of the oldest delta to be preserved.
−s	generates a shell procedure which reports *file* status.

comm selects or rejects lines common to two sorted files.

comm [**− 123**] *file1 file2*

1	contains lines only in *file1*.
2	contains lines only in *file2*.
3	contains lines in both files.

The minus sign "−" for a file name means standard input.

cp copies one file into another or into a directory.

cp *file1 file2*
cp *file directory*

cpio copies file archives in and out.

cpio **−o** [**aBcv**]
cpio **−i** [**Bdmrtuvs6**] [*patterns*]
cpio **−p** [**adlmuv**] *directory*

−o	copy out; obtains a list of path names from the standard input and copies those files onto the standard output with path name and status information.
−i	copy in; extracts from the standard input (**cpio** format) names of files that match *patterns*.
−p	pass; copies out and in with a single operation; destination path names are interpreted relative to the named *directory*.
a	resets access times of input files after they have been copied.
B	input/output is to be blocked 5,120 bytes to the record.
c	writes header information in ASCII characters for portability.
d	creates directories as needed.
l	links, rather than copies files whenever possible; usable only with the **−p** option.
m	retains previous file modification time.
r	interactively renames files.
s	swaps the bytes of words as they are read.
t	prints a table of contents of the input; creates no files.
u	copies unconditionally (overwrites).
v	verbose: prints a list of file names.
6	processes a UNIX Version 6 format file; only useful with **−i**.

cref makes a cross-reference listing of C programs and separates output in four columns: 1) symbol 2) filename 3) see below 4) text in file.

cref [**−ilnostux123**] *files*

i	the next argument is an ignore file.
l	puts line number in column 3 (instead of current symbol).
n	omits column 4 (no context).
o	the next argument is an only file.
s	current symbol in column 3 (default).
t	uses the next argument as the name of the intermediate file.
u	prints symbols that occur once.
x	prints C external symbols.
1	sorts output on column 1 (default).
2	sorts output on column 2.
3	sorts output on column 3.

crypt reads from files (or from the standard input) and writes to the standard output (or output file) using encode/decode passwords.

crypt [*password*] <*in.file* > *out.file*
crypt [*password*] <*out.file*> *in.file*

csh is a command interpreter with C-like syntax.

csh[– **cefinstvVxX**] [*file* ...]

–c	commands are read from the following required argument.
–e	exits the shell on an abnormal termination.
–f	ignores the .cshrc file.
–i	the shell is interactive and prompts even if it appears not to be a terminal.
–n	parses, but does not execute commands.
–s	takes command input from the standard input.
–t	reads and executes a single line of input.
–v	command input is echoed after history substitution.
–V	sets the verbose variable, even before .cshrc is executed.
–x	commands are echoed immediately before execution.
–X	sets the echo variable, even before .cshrc is executed.

csplit splits files according to contextual arguments.

csplit [**–s**] [**–k**] [**–f** *prefix*] *file arg1* [*...argn*]

–s	suppresses the printing of all character counts.
–k	leaves previously created files intact; default is removal.
–f *prefix*	names the created files *prefix00* ... *prefixn*; default is *xx00* ... *xxn*.

ct dials the telephone number of a modem attached to a terminal and spawns a login process.

ct [**–h**] [**–s***speed*] [**–v**] [**–w***n*] *telno*

–h	prevents hangup of current line if line is busy.
–s*speed*	specifies data transmission rate where *speed* is expressed in baud.
–v	sends a running narrative to standard error.
–w*n*	waits *n* minutes for an open line.
telno	is the telephone number.

ctags makes a tags file for **ex**(1) from the specified C or Fortran programs.

ctags [**–auw**] *file* ...

–a	appends output to the tags file instead of rewriting it.
–u	updates specified files in tags.
–w	suppresses warning diagnostics.

cu calls up another ZEUS system, a terminal, or a non-ZEUS system.

cu [**–s***speed*] [**–a***acu*] [**–l***line*] [**–h**] [**–e**| **–o**] *telno* | *dir*

–s*speed*	gives the transmission baud rate (110, 150, 300, 1200, 4800, 9600); 300 is the default value.
–a*acu*	specifies a device name for the ACU.
–l*line*	*line* is tty*X* line to be used for connection.
–h	emulates local echo, supporting calls to other computer systems which expect terminals to be in half-duplex mode.
–e(**–o**)	generates even (odd) parity for data sent to the remote system.
telno	is the telephone number.
dir	used for directly connected lines.

cut cuts out selected fields of each line of a file.

cut **–c***list* [*file1 file2* ...]
cut **–f***list* [**–d***char*] [**–s**] [*file1 file2* ...]

list	comma-separated list of integer field numbers (in increasing order), uses a dash (–) to indicate page ranges.
–c*list*	specifies character positions (e.g., **–c1–72** is the first 72 characters of each line).
–d*char*	the character following **–d** is the field delimiter (**–f** option only).
–f*list*	a *list* of fields separated by a delimiter character (e.g. **–f1,7** copies first and seventh fields).
–s	suppresses lines with no delimiter characters.

cw prepares constant-width text for **troff** when using the CW font.

cw [**–d**] [**–f***n*] [**–l***xx*] [**–r***xx*] [**–t**] [**+t**] *files*

–d	used for debugging; it prints current option settings on file descriptor 2 in the form of **troff**(1) comment lines.
–f*n*	mounts CW font in *n* position.
–l*xx*	is a one- or two-character string which defines the left delimiter.
–r*xx*	same as above, for the right delimiter.
–t	turns transparent mode off.
+t	turns transparent mode on (default).

cxref lists routines in a C program.

cxref *file* ...

date prints the current date and time.

date [**–u**]

–u	prints GMT time.

daytime prints in English the current time of day, accurate to the nearest five minutes.

daytime

dc is a desk calculator; an arbitrary precision stack-structured arithmetic package. See **dc**(1) for constructions.

dc [*file*]

dd converts and copies a file.

dd [*option* = *value*] ...

Options	Values
bs=*n*	sets both input and output block size.
cbs=*n*	conversion buffer size.
conv=ascii	converts EBCDIC to ASCII.
ebcdic	converts ASCII to EBCDIC.
ibm	slightly different map of ASCII to EBCDIC.
lcase	maps alphabetics to lowercase.
noerror	does not stop processing on an error.
swab	swaps every pair of bytes.
sync	pads every input record to **ibs.**
ucase	maps alphabetics to uppercase.
... , ...	several comma-separated conversions.
count=*n*	copies only *n* input records.
files=*n*	skips *n* files before starting copy.
seek=*n*	seeks *n* records from beginning of output file before copying.
ibs=*n*	input block size *n* bytes (default 512).
if=*file*	input file name; standard input is default.
obs=*n*	output block size (default 512).
of=*file*	output file name; standard output is default.
skip=*n*	skips *n* input records before starting copy.

delta makes a delta (change) to an SCCS file.

delta [**−r***SID*]
[**−g***list*]
[**−m**[*mrlist*]]
[**−n**]
[**−p**]
[**−s**]
[**−y**[*comment*]] *files*

−r*SID*	specifies which delta is to be made to the SCCS file.
−g*list*	*list* of deltas to be ignored when the file is accessed at the change level (SID) created by this delta.
−m[*mrlist*]	inserts Modification Request (*MR*) numbers into the SCCS file.
−n	retains the edited *g-file.*
−p	prints (on the standard output) the SCCS file differences before and after the delta is applied in a **diff**(1) format.
−s	suppresses output of status information.
−y[*comment*]	text describing the reason for making the delta.

deroff removes **nroff/troff, tbl,** and **eqn** constructs.

deroff [**−mx**] [**−w**] [*files*]

−mx	**−mm** or **−ms** options suppresses text from macro lines; the **−ml** option forces the **−mm** option and deletes lists associated with the **MM** macros.
−w	outputs a word list, one "word" per line.

diff lists differing lines between two files.

diff [**−befh**] *file1 file2*

−b	ignores trailing tabs and spaces in the comparison.
−e	produces a script of **a, c,** and **d** commands for the editor **ed**(1), which recreates *file2* from *file1*.
−f	produces a similar script, in reverse order; not useful with **ed**(1).
−h	works when changed parts are short and well separated. Options **−e** and **−f** are illegal with **−h.**

diff3 compares 3 files and lists differences.

diff3 [**−ex3**] *file1 file2 file3*

−e	**diff3** outputs an **ed**(1) script that incorporates into *file1* changes between *file2* and *file3.*
−x (**−3**)	produces a script to incorporate only changes flagged "= = = =" ("= = = =3").

diffmk compares two versions of a file and creates a third that includes "change mark" commands for **nroff**(1) or **troff**(1).

diffmk *name1 name2 name3*

dircmp compares dir1 and dir2, generating information about the differences between the directories.

dircmp *dir1 dir2*

dirname (see **basename** entry) delivers all but the last level of a pathname in string.

dirname *string*

dog is a text filter for CRT previewing.

dog [*file*...]

du summarizes disk usage.

du [**−ars**] [*files*]

−a	generates an entry for each file.

−r	generates messages when files and directories cannot be opened or read.
−s	gives the grand total for each of the specified *files*.

echo is both an internal shell command, and an external program; it writes arguments separated by blanks and terminated by a newline on the standard output.

echo [**−n**] [*arg*] ...

−n	no newline is added to the output.

echo2 echos (prints) arguments to the standard error.

echo2 [**−n**] [*arg*] ...

−n	no newline is added to the output.

ed is the standard line-oriented text editor.

ed [**−**] [**−x**] [*file*]

−	suppresses the printing of character counts on **e**, **r**, and **w** commands.
−x	an **x** command is simulated first to handle an encrypted file.

edit is a variant of the text editor **ex**(1) recommended for new or casual users.

edit [**−r**] *file* ...

−r	recovers files after an editor or system crash; the last saved version is retrieved.

egrep searches a file for full regular expressions (see **grep**).

egrep [*options*] [*expression*] [*files*]

env sets environment for command execution.

env [**−**] [*name*=*value*] ... [*command args*]

−	ignores inherited environment; executes a command with the environment specified by the arguments.

eqn is a **troff** preprocessor for typesetting mathematics on a phototypesetter.

eqn [**−d***xy*] [**−f***n*] [**−p***n*] [**−s***n*] [*file*] ...

−d*xy*	sets delimiters to *x* and *y*.
−f*n*	changes font *n*.
−p*n*	changes sub- and superscript point size.
−s*n*	changes size *n*.

error analyzes and disperses compiler error messages to the source file and line where the errors occurred.

error [**−I** *ignorefile*] [**−n**] [**−q**] [**−s**] [**−t** *suffix-list*] [**−v**] [*file*]

−I *ignorefile*	names file containing the names of the functions to ignore.

−n	does not touch any files; all error messages are sent to the standard output.
−q	queries the user whether to touch the file.
−s	prints statistics of error categorization.
−t	the following argument is a *suffix-list*.
−v	after touching files, executes **vi**(1).

ex is the root of a family of editors: **edit**, **ex**, and **vi**. **Ex** is a superset of **ed**, with display editing.

```
ex [ − ]
   [ −l ]
   [ −r ]
   [ −R ]
   [ −ttag ]
   [ −v ]
   [ +[command ]]
```

edit [*ex options*] *file*...

−	suppresses editor prompts and character counts.
−l	sets up **ex** for Showmatch and Lisp options.
−r	recovers files after an editor or system crash.
−R	invokes a "read only" version of **ex**.
−t*tag*	positions the cursor at *tag* when **ex** is entered.
−v	invokes **vi** instead of **ex**.
+[*command*]	the editor begins by executing the *command*.

expand changes tabs to spaces and writes to the standard output.

expand [**−***tabstop*] [**−***tab1,tab2,...tabn*] [*file* ...]

expr evaluates arguments as an expression.

expr *arg* ...

fgrep searches file for fixed strings (see **grep**).

fgrep [*options*] [*strings*] [*files*]

file determines file type.

file [**−f**] *file* ...

−f	next argument is a *file* containing names of files to be examined.

find recursively descends the directory hierarchy for each pathname seeking files that match a Boolean expression.

find *pathname-list expression*

−atime *n*	true if files have been accessed in *n* days.
−cpio *device*	write the current file on *device* in **cpio**(5) format (5120 byte records).
−ctime *n*	true if the file has been changed in *n* days.

−exec *cmd*	true if the executed *cmd* returns a zero value as exit status.
−group *gname*	true if the file belongs to the group *gname*.
−links *n*	true if the file has *n* links.
−mtime *n*	true if the file has been modified in *n* days.
−name *file*	true if *file* matches the current file name.
−newer *file*	true if the current file has been modified more recently than the argument *file*.
−ok *cmd*	the command line is echoed with a question mark, and is executed if the user types **y**.
−perm *onum*	true if the file permission flags exactly match the octal number *onum* (see **chmod**(1)).
−print	always true; prints the current pathname.
−size *n*	true if the file is *n* blocks long (512 bytes per block).
−type *c*	true if the type of the file is *c*, where *c* is **b**, **c**, **d**, **p**, or **f** for block special file, character special file, directory, fifo (a.k.a. pipe), or plain file.
−user *uname*	true if the file belongs to the user *uname*.
(*expression*)	true if the parenthesized expression is true.

flow performs a flow analysis of C programs.

flow [**−bcors**] [*output-suffix*] *files* ...

−b	generates the 'CALLEDBY' file.
−c	generates the 'CALLS' table.
−o	uses a suffix supplied by the user for output tables (the next argument).
−r	generates the 'RESIDES' file.
−s	saves the trace files.

get generates an SCCS text file by keyletter arguments, which begin with −.

get [−*aseq*] *file*
[−**b**]
[−*ccutoff*]
[−**e**]
[−**i***list*]
[−**k**]
[−**l**[**p**]]
[−**r***SID*]
[−**x***list*]
[−**bgmnpst**] *file*

−*aseq*	retrieves a specified delta *sequence* number.

−b	with **−e**, indicates the new delta should have an SID in a new branch.
−*ccutoff*	provides a *cutoff* date-time for changes, in the form: YY[MM[DD[HH[MM[SS]]]]]
−e	gets a file for editing or making a change (delta).
−**i***list*	*list* of deltas to include in creating the generated file.
−k	suppresses keyword replacement in the retrieved text.
−l[**p**]	writes a delta summary to an l.*file*; **−lp** writes only to the standard output.
−**r***SID*	specifies the *SID* string of the version (delta) of an SCCS file to be retrieved.
−**x***list*	*list* of deltas to exclude in creating the generated file.
−g	suppresses retrieval of text from the SCCS file; used for SID verification.
−m	precedes each text line retrieved from the SCCS file with the SID of the delta that inserted it in the SCCS file.
−n	precedes text lines with the %M% identification keyword value.
−p	prints text retrieved from the SCCS file to the standard output.
−s	suppresses output normally written on the standard output.
−t	accesses the most recent ("top") delta in a given release.

getfile transfers files from local to remote systems.

getfile [−**bBfq**] *file1* [[−**b**] *file2* ...]

−b	the next file is binary. Carriage returns are not replaced by new lines.
−B	all files are treated as if they were preceded by a **−b**. Desirable for ZEUS-to-ZEUS transfers.
−f	suppresses all nonfatal error messages.
−q	queries before overwriting files.

getNAME gets NAME sections of manual for **whatis/apropos** data base.

getNAME *name* ...

gpasswd changes or installs a group password.

gpasswd [*name*]

greek filters the extended character set of a Model 37 terminal for other terminal types.

greek [**−T***terminal*]

grep searches a file for a pattern.

grep [*options*] *expression* [*files*]

−b	each line is preceded by its block number.
−c	prints a count of matching lines.
−e *expression*	used when the *expression* begins with a −.
−f *file*	the regular *expression* (**egrep**) or *strings* list (**fgrep**) is taken from *file*.
−h	does not print filename headers with output lines.
−l	names only files with matching lines.
−n	each line is preceded by its relative line number in the file.
−s	suppresses error messages for nonexistent or unreadable files (**grep** only).
−v	prints all lines but those that match.
−x	prints only lines exactly matched (**fgrep** only).

grpck is a group file checker.

grpck *file*

hd dumps file in hex (see **od**).

hd [**−bcdox**] [*file*] [[**+** [**x**]] *offset* [**.**] [**b**]]

head prints the first few lines of a file.

head [− *count*] [*file* ...]

−*count*	specifies line *count*; default is 10.

help is online information explaining commands and their messages.

help [*args*]

hyphen finds and prints hyphenated words.

hyphen *files*

id prints user and group IDs and names.

id

isrio determines if terminal is a RIO system.

isrio

join forms, on the standard output, a merge of the two relations specified by the lines of file1 and file2.

join [*options*] *file1 file2*

−a*n*	with normal output, produces a line for each unpairable line in file *n*, where *n* is **1** or **2.**
−e *s*	replaces empty output fields by string *s*.
−j*n m*	join on the *m*th field of file *n*; if *n* is missing, use the *m*th field in each file.
−o *list*	each output line comprises the fields specified in *list*, each element of which has the form *n.m*, where *n* is a file number and *m* is a field number.
−tc	uses character c as a separator (tab character).

kill kills specified processes.

kill [−*signo*] *processid* ...

1	hangup
2	interrupt
3*	quit
4*	illegal instruction
5*	trace trap
6*	IOT
7*	EMT
8*	floating exception
9	kill
10*	bus error
11*	segment violation
12*	bad system call
13	write on pipe with no one to read
14	alarm clock
15	software terminate
16	unassigned
*	causes core dump

ld creates nonsegmented load modules for execution under ZEUS and for downloading to target hardware.

ld [*option*] *file* ...

−b *addr*	
−bx *addr*	sets the bottom location for the program, or for the specified section of *x*. *X* can be **t**, **d**, or **b**, for text, data or bss.
−d	defines common storage even if using the **−r** flag.
−e *name*	the following argument names the entry point of the loaded program.
−i	separates program text and data areas when the output file is executed.
−lx	searches the named library identified by string *x*.
−Nx	specifies number of characters for use in symbol table name entry.
−o *name*	changes **ld** output filename to *name*.
−r	generates relocation bits in the output file for use in subsequent **ld** runs.
−s	strip the output: removes symbol table and relocation bits to save space.
−t *addr*	
−tx *addr*	sets the highest location of the program or section to the hex, octal, or decimal number specified. *X* can be **t**, **d**, or **b**, for text, data or bss.
−u *symbol*	enter *symbol* into the symbol table as undefined.
−w	suppresses symbol redefinition warning.

−x	enters only external and global symbols; doesn't preserve local symbols in the output symbol table.
−X	saves local symbols except for section name entries and those beginning with L in the symbol table.

learn is an on-line computer-aided instruction package.

learn [*subject* [*lesson*]]

lex generates programs to be used in simple lexical analysis of text.

lex [**−fntv**] [*file*] ...

−f	faster compilation; limited to small programs.
−n	opposite of **−v**; **−n** is default.
−t	places the result on standard output; default is lex.yy.c.
−v	prints a summary of statistics from the generated analyzer.

line reads one line (up to a new-line) from the standard input and writes it on the standard output.

line

lint checks C programs for programming errors and bad practices.

lint [**−abchnpuvx**] *file* ...

−a	reports assignments of long values to int variables.
−b	reports break statements that cannot be reached.
−c	complains about casts with questionable portability.
−h	applies heuristic tests to intuit bugs, improve style, and reduce waste.
−n	do not check compatibility against the standard library.
−p	checks portability to the IBM and GCOS dialects of C.
−u	do not complain about functions and variables used and not defined, or defined and not used.
−v	suppresses complaints about unused arguments in functions.
−x	reports variables referred to by extern declarations, but never used.

ln links a filename to an actual file.

ln *name1 name2*

LOAD downloads the text, data, and bss sections into the Z8000 or Z8 Development Module (DM) or ICP 8/02 (communications processor).

LOAD *file*

local returns control to the local system.

local [**−l**]

−l	gives a "logout" to the remote system before returning to the local system.

login is a sign-on command to the computer.

−name	bypasses the message-of-the-day display.

logname gets a login name.

logname

look prints all lines in a sorted list that begin with a designated string.

look [**−df**] *string* [*file*]

−d	dictionary order: only letters, digits, tabs, and blanks are compared.
−f	fold: uppercase letters compare equal to lowercase.

lorder finds loading order of an object library.

lorder *file* ...

lpr is a line printer spooler.

lpr [*option*] ... [*file*] ...

Refer to **nq**(1) for more information.

ls lists the contents of a directory.

ls [**−aAcCdDfFgilmnqrRstuxl**] *file* ...

−a	lists all entries, including .files.
−A	like **−a**, but . and .. are suppressed.
−c	lists time of last change to i-node.
−C	forces multicolumn output; default for CRT output.
−d	lists directory name, not contents.
−D	lists only directories.
−f	forces each argument to be interpreted as a directory and lists the names found in each slot.
−F	indicates directories by appending a / to the filename, and executable files by appending a *.
−g	gives group ID in long listing; use only with the **−l** function.
−i	prints i-numbers before each file listing.
−l	for each file, lists in long format, permission mode bits, number of links, owner, group, size in bytes, and time of last modification.
−m	forces stream output format.
−n	in long listings, gives numeric uid or group id.

−q	forces printing of nongraphic characters in file names as the character ? (default if output device is a terminal).
−r	reverses the order of sort to get reverse alphabetic or oldest first.
−R	recursively lists the contents of each directory found.
−s	gives size in blocks, including indirect blocks, for each entry.
−t	sorts by time modified (latest first) instead of by name.
−u	uses time of last access for sorting (**−t**) or printing (**−l**).
−x	forces columnar printing to be sorted across the page.
−1	forces one entry per line output format.

m4 is a macro processor intended as a front end for Ratfor, C, and other languages.

m4 [*files*]

mail sends and receives mail among users.

mail *persons*

mail [**−f** *file*] [**−pqr**]

−f *file*	uses *file* (e.g., mbox) instead of the default *mail* file, $MAIL.
−p	prints mail without prompting for disposition.
−q	terminates after interrupt.
−r	prints messages with first-in, first-out order.

make maintains, updates, and regenerates groups of programs.

make [**−bdeiknpqrst**] [**−f** *makefile*] [*files*]

−b	compatibility mode for old makefiles.
−d	debug mode. Prints information about files and times examined.
−e	environment variables override assignments within makefiles.
−f *makefile*	*makefile* is the name of a description file. A file name of − denotes the standard input. The contents of *makefile* override the built-in rules if they are present.
−i	ignores error codes returned by invoked commands. This mode is entered if the fake target name **.IGNORE** appears in the description file.
−k	abandons work on the current entry, but continues on other branches that do not depend on that entry.
−n	prints commands, but does not execute them; lines beginning with an @ are printed.
−p	prints the complete set of macro definitions and target descriptions.
−q	question. The **make** command returns a zero or non-zero status code depending on whether the target file is or is not up-to-date.
−r	does not use the built-in rules.
−s	silent mode. Does not print command lines before executing. This mode is also entered if the target name **.SILENT** appears in the description file.
−t	touch the target files (causing them to be up-to-date) rather than issue the usual commands.
.DEFAULT	if a file must be made but there are no explicit commands or relevant built-in rules, the commands associated with the name **.DEFAULT** are used if it exists.
.IGNORE	same effect as the **−i** option.
.PRECIOUS	dependents of this target will not be removed when quit or interrupt are hit.
.SILENT	same effect as the **−s** option.

man prints the section of the ZEUS Reference Manual by named title in the specified chapter.

man [*option* ...] [*chapter*] *title* ...

−e	preprocesses with **neqn** or **eqn**(1); **−e** alone means **−te.**
−h	sends to the output device using **cat;** default is **more**. Intended primarily for hardcopy terminals.
−n	prints the section on the standard output using **nroff**(1).
−t	phototypesets the section using **troff**(1).
−v	sends to the output device using **view**.
−w	prints the path names of the manual sections, but does not print the sections themselves.
(default)	copies formatted manual section to the terminal, or, if none is available, act as **−n**. It may be necessary to use a filter to adapt the output to the terminal's characteristics.

mesg permits or denies messages coming in to a terminal.

mesg [**n**] [**y**]

n	forbids messages.
y	permits messages.

mkdir creates directories.

mkdir *dirname* ...

mkmenu is a menu creation utility used by **zmenu**(1).

mkmenu [*file*]

mkstr creates an error message file by massaging C source.

mkstr [–] *messagefile prefix file* ...

–	places the error messages at the end of the specified *messagefile* when recompiling part of a large **mkstr** program.

mm prints documents formatted with the **MM** macros.

mm [*options*] [*files*]

–12	indicates a 12 pitch printing.
–c	invokes **col**(1).
–e	invokes **neqn**(1).
–E	invokes the **nroff**(1) **–e** option.
–t	invokes **tbl**(1).
–t*Term*	specifies the type of output terminal.
–y	uses a non-compacted version **mm**(7).

mmchek checks usage of **MM** macros and **eqn** delimiters.

mmchek [*files*]

mmt uses the **MM** macro package with **troff**(1) to typeset documents.

mmt [*options*] [*files*]

–a	invokes **–a** option of **troff**(1).
–e	invokes **eqn**(1).
–t	invokes **tbl**(1).
–y	**mmt** uses the non-compacted macros **mm**(7).

more examines a file on a terminal one screenful at a time.

more [**–dfl***n*] [+*line*] [+*/pat*] [*file* ...]

–d	prompts the user with the message "Hit space to continue, Rubout to abort" at the end of each screenful.
–f	counts logical, rather than screen lines; long lines are not folded.
–l	does not treat ^L (form feed) specially.
n	defines the number of lines in a window.
+*line*	starts at *line*.
+*/pat*	starts two lines before the line containing the regular expression *pat*tern.

mv moves (renames) files and directories.

mv *file1 file2*

mv *file ... directory*

mvt uses the macro package for view graphs and slides with **troff** for typesetting (see **mmt** for options).

mvt [*options*] [*files*]

neqn (see **eqn** entry) is a **troff**(1) preprocessor for typesetting mathematics on terminals.

neqn [**–d***xy*] [**–f***n*] [**–p***n*] [**–s***n*] [*file*] ...

newgrp logs into a new group.

newgrp [*group*]

news informs the user of current events.

news [**–ans**] [*items*]

–a	prints all items, regardless of age; the stored time is not changed.
–n	reports names of current items without printing their contents, and without changing the stored time.
–s	reports how many current items exist, without printing their names or contents, and without changing the stored time.

nice executes a command with low scheduling priority.

nice [**–***number*] *command* [*arguments*]

nl is a line numbering filter.

nl [**–b***type*] [**–f***type*] [**–h***type*] [**–i***incr*] [**–l***num*] [**–n***format*] [**–p**] [**–s***sep*] [**–v***start#*] [**–w***width*] *file*

–b*type*	specifies which logical page body lines are to be numbered. Recognized *types* are: **a**, number all lines; **t**, number lines with printable text only; **n**, no line numbering; **p***string*, number only lines that contain the expression *string*. Default *type* for logical page body is **t**.
–f*type*	same as **–b***type* except for footer. Default for logical page footer is **n**.
–h*type*	same as **–b***type* except for header. Default *type* for logical page header is **n**.
–i*incr*	*incr* is the increment value used to number logical page lines. Default is 1.
–l*num*	*num* is the number of blank lines to be considered as one. For example, **–l2** results in only the second adjacent blank being numbered (if the appropriate **–ha**, **–ba**, and/or **–fa** option is set). Default is **l1**.
–n*format*	*format* is the line numbering format. Recognized values are: **ln**, left justified, leading zeroes suppressed; **rn**, right justified, leading zeroes suppressed; **rz**, right justified, leading zeroes kept. Default *format* is **rn**.
–p	do not restart numbering at logical page delimiters.

-s*sep*	*sep* is the character(s) used in separating the line number and the corresponding text line. Default *sep* is a tab.
-v*start#*	*start#* is the initial value used to number logical page lines. Default is **1.**
-w*width*	*width* is the number of characters for the line number. Default is 6.

nm prints the name list (symbol table) of each file specified. a.out is the default file.

File Section
Data Bss Common
Undef Absolute Text

nm [**-gnoprsu**] [*file*] ...

-g	prints only global (external) symbols.
-n	sorts numerically rather than alphabetically.
-o	prepends file or archive element name to each output line.
-p	prints in symbol-table order rather than in sorted order.
-r	sorts in reverse order.
-s	sorts according to the size of the external symbol.
-u	prints only undefined symbols.

nohup executes a command with telephone hangups and quits ignored.

nohup *command*

nq is a general purpose enqueuing program.

nq [*option*] [*file*] ...

-b	returns to burst page printing (default) if turned off by a previous option.
-c	copies the file, protecting it against changes that can occur before printing occurs.
-d *dest*	sets the destination field. With **-m** specified, and if *dest* is a valid login name, also sends **mail**(1) to user *dest*.
-m	reports by **mail**(1) when printing is complete.
-n[bcdmpqrst]	negates previous options. For the letter given, resets option to the default value; default is **-nm.**
-p *pri*	file is printed at priority *pri*. Valid priorities are **normal**, **deferred**, and **rush**. Accepts abbreviations to one character.
-q *que*[:*dev*]	sends the files to queue *que* for printing on the first available device. Can request a specific device with the :*dev* option.
-r	removes the file after printing.
-s	silence. Prevents file name from appearing in queue lists (see **xq**(1)).
-t *number*	prints file *number* times.

nroff formats text for terminal or line printer output (see **troff** for additional options).

nroff [**-cikmnoqrsehTz**] ... [*file*] ...

Nroff Only

-e	produces equally-spaced words in adjusted lines, using full terminal resolution.
-h	uses output tabs during horizontal spacing to speed output and reduce output character count.
-T*name*	prepares output for specified terminal.
-z	prints messages generated by **.tm** (terminal message) requests.

objdu is a hex dump for object and load modules.

objdu [**-h**] [**-r**] *file*

-h	prints only the header information.
-r	prints relocation information if the file is not stripped.

objhdr dumps System 8000 load module header information in hex format.

objhdr *file*

objsu strips the leading underscore from names in the symbol table section of a.out files.

objsu *file* ...

od is an octal dump.

od [**-bcdox**] [*file*] [[**+** [**x**]] *offset* [**.**] [**b**]]

-b	interprets bytes in octal.
-c	interprets bytes in ASCII.
-d	interprets words in decimal.
-o	interprets words in octal.
-x	interprets words in hex. Displays addresses in hex.

pack compresses files.

pack [**-**] *file* ...

page examines a file on a terminal one screenful at a time (see **more**).

page [**-dfln**] [+*line*] [+/*pat*] [*file* ...]

passwd changes login password.

passwd [*name*]

paste merges same lines of several files or subsequent lines of one file.

paste *file1 file2* ...
paste **-d***list file1 file2* ...
paste **-s** [**-d***list*] *file1 file2* ...

-d*list*	replaces the tab character with character(s) in *list*.
-s	merges subsequent lines rather than one from each input file.
-	can be used in place of any file name to read a line from the standard input; there is no prompting.

pcat expands files (see **pack**).

pcat *file* ...

pr prints the named files on the standard output.

pr [**adefhilmnoprstw+ –**] [*files*]

–a	prints multi-column output across the page.
–d	double-spaces the output.
–e*ck*	expands *input* tabs to character positions *k*+1, 2**k*+1, 3**k*+1, etc. *c* is any non-digit character; default is tab.
–f	uses form-feed character for new pages (default uses a sequence of line-feeds).
–h	uses the next argument as the header to be printed instead of the file name.
–i*ck*	in *output*, replaces white space wherever possible by inserting tabs to character positions *k*+1, 2**k*+1, 3**k*+1, etc.
+*k*	begins printing with page *k* (default is **1**).
–*k*	produces *k*-column output (default is **1**). **–e** and **–i** are assumed for multi-column output.
–l*k*	sets the page length to *k* lines (default is **66**).
–m	merges and prints all files simultaneously, one per column (overrides the –*k*, and **–a** options).
–n*ck*	provides *k*-digit line numbering (default is **5**).
–o*k*	offsets each line by *k* character positions (default is **0**).
–p	pauses before beginning each page if the output is directed to a terminal (**pr** will ring the bell at the terminal and wait for a carriage return).
–r	prints no diagnostic reports on failure to open files.
–s*c*	separates columns by the single character *c* instead of by the appropriate number of spaces (default for *c* is a **tab**).
–t	do not print the five-line header or the five-line footer for each page. Quits printing after the last line of each file without spacing to the end of the page.
–w*k*	sets line width to *k* characters (default is **72**).

printenv displays environment variables.

printenv [*variable*]

prof displays profile data.

prof [**–al**] [*file*]

–a	reports all symbols, not just external symbols.
–l	lists output by symbol value rather than decreasing percentage.

prom outputs a file to a PROM programming device.

prom [**–b**] [**–d** *device*] [**–Dlos** *hex*] *file*

–b	breaks word-oriented files into high and low bytes so that 16-bit data can be sent to two 8-bit PROMS.
–d	selects output device by pathname for either a device or file.
–D	for separate I and D files, outputs only the data section.
–l	sets the PROM length for transfer to designated device; default is **0x800.**
–o	sets the offset (beginning address) of the first data to be transferred; default is **0x000.**
–s	selects the segment number from the input file to be programmed; default is **0x00.**

prs prints, on the standard output, parts or all of an SCCS file.

prs	[**–a**] *files* [**–d**[*dataspec*]] [**–e**] [**–l**] [**–r**[*SID*]] *files*
–a	requests printing of information for both removed (delta type = R, see **rmdel**(1)) and existing (delta type = D) deltas.
–d[*dataspec*]	specifies the output data. The *dataspec* is a string consisting of SCCS file data keywords interspersed with optional user supplied text.
–e	requests information for all deltas created earlier than and including the delta designated with the **–r** keyletter.
–l	requests information for all deltas created later than and including the delta designated with the **–r** keyletter.
–r[*SID*]	used to specify the SCCS *ID*entification (*SID*) string of a delta for which information is desired. If no SID is specified, the most recently created delta is assumed.

ps prints information about active processes.

ps [**−adefl**] [**−g** *glist*] [**−n** *namelist*] [**−p** *plist*] [**−t** *tlist*] [**−u** *ulist*]

−a	prints information about all processes except process group leaders and processes not associated with a terminal.
−d	prints information about all processes, except process group leaders.
−e	prints information about all processes.
−f	generates a full listing.
−g *glist*	restricts listing to data about processes whose process groups are given in *glist*.
−l	generates a long listing.
−n *namelist*	the argument is taken as the name of an alternate *namelist* (**/zeus** is the default).
−p *plist*	restricts listing to data about processes whose process ID numbers are given in *plist*, where *plist* is in the same format as *tlist*.
−t *tlist*	restricts listing to data about the processes associated with the terminals given in *tlist*.
−u *ulist*	restricts listing to data about processes whose user ID numbers or login names are given in *ulist*.

ptx generates a permuted index.

ptx [**−bfgiortw**] ... [*input* [*output*]]

−b *break*	uses the characters in the *break* file to separate words; tab, newline, and space characters are used as break characters.
−f	folds upper- and lowercase letters for sorting.
−g *n*	uses *n* as the number of characters allowed for each gap among the four parts of the line as finally printed. The default gap is 3 characters.
−i *ignore*	do not use as keywords any words given in the *ignore* file. If the **−i** and **−o** options are missing, use /usr/lib/cref/eign as the *ignore* file.
−o *only*	uses as keywords words given in the *only* file.
−r	takes any leading nonblank characters of each input line to be a reference identifier (as to a page or chapter) separate from the text of the line. Attaches that identifier as a 5th field on each output line.

−t	prepares the output for the phototypesetter; the default line length is 100 characters.
−w *n*	uses *n* as the width of the output line. The default line length is 72 characters.

putfile downloads files from a remote ZEUS to a local system running ZEUS or RIO.

putfile [**−bBfq**] *filename1* [[**−b**] *filename2* ...]

−b	the next file is considered to be a binary. Creates a binary file on the local system; new lines are not replaced by carriage returns.
−B	treats all file names on the line as if they are preceded by a **−b**; desirable for a ZEUS-to-ZEUS transfer.
−f	suppresses all nonfatal error messages.
−q	queries before overwriting a file.

pwck is a password file checker.

pwck [*file*]

pwd prints the pathname of the present working directory.

pwd

ranlib converts archives to random libraries.

ranlib *archive* ...

regcmp is a regular expression compiler.

regcmp [−] *files*

−	places output in *file*.c.

remote transfers control to a remote system running ZEUS or UNIX software.

remote [*system name*]

reserv provides a mechanism for reserving a system tape drive.

reserv [**−fimNqrN**]

−f	frees the tape drive. Only the user who originally reserved the tape drive can free it.
−i*minutes*	specifies an interval in *minutes* that **reserv** waits since last access to free tape.
−m0, ..., m7	selects mag tape drive number; default is 0.
−q	queries drive to find out if it is busy.
−r	requests to be put on the queue of users waiting for the tape drive if tape drive is busy.
−0, ..., 7	selects cartridge tape drive number; default is 0.

reset resets terminal modes to default values.

reset<linefeed>

rm removes (unlinks) one or more files from a directory.

rm [**-fir**] *file* ...

-f	no questions are asked with the **-f** (force) option.
-i	asks whether to delete each file (interactive option), and, under **-r**, whether to examine each directory.
-r	recursively deletes the entire contents of the specified directory, and the directory itself.

rmail allows mail to be sent only.

rmail *persons*

rmdel removes the delta specified by the SID from each named SCCS file.

rmdel -r*SID files*

-r*SID*	specifies which delta is to be removed from the SCCS file.

rmdir removes directories (which must be empty, see **rm**).

rmdir *dir*

rsh is a restricted version of the standard command interpreter **sh**(1).

rsh [*flags*] [*name*[*arg1*...]]

flags	refer to **sh**(1) for flag definition.

sact prints current SCCS file editing activity.

sact *files*

scc is the System 8000 segmented C compiler.

scc [**-cDEIOpPSU**] ... *file* ...

-c	compiles and assembles the named C source files but suppresses the linking step; forces an object file .o to be produced.
-D*name*	
-D*name*=*def*	defines *name* to the preprocessor, as if by #define. If no definition is given, *name* is defined as 1.
-E	runs only the macro preprocessor and sends the result to the standard output.
-I*dir*	brings in a directory of #include files.
-O	invokes the C optimizer for System 8000 code.
-p	arranges for the compiler to produce code that counts the number of times each routine is called.
-P	runs only the macro preprocessor and places the result for each .c file in a corresponding .i file with no # lines in it.
-S[l]	compiles the named C source files but suppresses the assembly and link step. Leaves the assembly language code in corresponding files suffixed with .s. If l is specified, the original C source lines appear as assembly language comments preceding the code produced for them.
-U*name*	removes any initial definition of *name*.

sccsdiff compares two versions of an SCCS file and generates the differences between the two versions.

sccsdiff -r*SID1* **-r***SID2* [**-p**] [**-s***n*] *file*...

-p	pipes output for each file through **pr**(1).
-r*SID?*	*SID1* and *SID2* specify the deltas of an SCCS file that are to be compared. Versions are passed to **bdiff**(1) in the order given.
-s*n*	*n* is the file segment size that **bdiff** will pass to **diff**(1). This is useful when **diff** fails due to a high system load.

script makes a file copy of all terminal interactions.

script [**-a**] [**-q**] [**-S** *shell*] [*file*]

-a	appends output to the file named **typescript** instead of creating a new file.
-q	quiet mode: turns off the "script started" and "script done" messages.
-S *shell*	shell specifier. The default *shell* is specified in the user's entry of the /etc/passwd file. If the requested shell is not available, **script** uses any shell it can find.

sdiff uses the output of **diff**(1) to produce a side-by-side listing of two files indicating lines that are different.

sdiff [**-losw** ...] *file1 file2*

-l	prints the left side of lines that are identical.
-o *output*	names *output* as the third file that is created as a user controlled merging of *file1* and *file2*.
-s	does not print identical lines.
-w *n*	*n* defines the width of the output line; the default is 130 characters.

sed copies files (standard input default) to the standard output, edited according to a script of commands.

sed [**-e** *script*] [**-f** *sfile*] [**-n**] [*file*] ...

-e *script*	If there is just one **-e** option and no **-f**'s, **-e** can be omitted.
-f *sfile*	takes the script from file *sfile*; these options accumulate.
-n	suppresses the default output.

SEND is invoked from the monitor and uploads memory from a Zilog Development Module (DM) to the ZEUS system.

SEND *file start end* [*entry*]

sh (shell) is the Bourne shell command programming language that executes commands from a terminal or a file.

sh [**−ceiknrstuvx**] [*args*]

−c *string*	commands are read from *string.*
−e	for non-interactive shell, exit on bad status.
−i	this shell is interactive.
−k	keyword arguments are placed in the command environment.
−n	reads commands but does not execute them.
−r	the shell is restricted.
−s	reads commands from the standard input. Arguments specify the positional parameters. Shell output is written to file descriptor 2.
−t	exits after reading and executing one command.
−u	treats unset variables as errors when substituting.
−v	prints line before executing it.
−x	prints commands and arguments as executed.
− −	does not change any of the flags; useful in setting $1 to -.

size prints the size (in bytes) of the text, data and bss sections of object files.

size [*file* ...]

sld links several Z8000 object programs into one segmented load module, resolving external references, and searching libraries.

sld [**−AbdefilMNOoprRstuwxXz**] ... *file* ...

−Ap *number*	changes space allocation for pass one segment table entries.
−As *number*	changes space allocation for symbol table.
−b *addr*	
−bx *addr*	sets the bottom location for the program, or for the specified section if *x* is specified. *x* can be one of **t**, **d**, or **b** for text, data, and bss, respectively.
−d	forces definition of common storage, even if the **−r** flag is present.
−e *name*	names the entry point of the loaded program.
−f *file*	uses the contents of *file* for an argument list.

−i	separates the program text and data areas when the output file is executed.
−l*x*	searches the named library; *x* is a string to be substituted in /lib/slib*x*.a.
−M*y*	
−M*xy*	
−M*y* + *file* ... +	
−M*xy* + *file* ... +	
−Ms*y*	specifies section to segment mappings. *x* can be one of **t**, **d**, or **b** for text, data and bss respectively. *y* is a decimal integer, between 0 and 127 (inclusive), which indicates target segment.
−N*x*	specifies the number of characters in the name of a symbol table entry.
−o *name*	changes the name of the **sld** output file to the name specified.
−r	generates relocation bits in the output file for use in subsequent **sld** runs.
−R*y*	creates a zero length segment.
−s	removes the symbol table and relocation bits to save space.
−t *addr*	
−t*x* *addr*	sets the highest location of the program or section to the hex, octal, or decimal number specified. *x* can be one of **t**, **d**, or **b** for text, data, and bss.
−u	takes the following argument as a symbol and enters it as undefined in the symbol table.
−w	suppresses the symbol redefinition warning.
−x	enters only external and global symbols.
−X	saves local symbols except those beginning with L.
−z	gives the output file default mappings for the System 8000.

sleep suspends execution for specified time in seconds.

sleep *time*

slink creates load modules for downloading to target hardware.

slink [**−eioPsvwxX**] [[**−S** *addr*] *file* ...] ...

−e *name*	*name* is the entry point of the program. The link address of the text section is the default.
−i	separates the program text and data areas when the output file is executed; Z8 object modules only.
−o *name*	changes the name of the **slink** output file to the *name* specified.
−P	preserves the temporary work files in /tmp/sl...... for debug purposes.

−s	removes the symbol table and relocation bits to save space.
−v	verbose option. Prints the command line passed to **ld**.
−w	suppresses symbol redefinition warnings produced while searching archives.
−x	enters only external and global symbols.
−X	saves local symbols in the symbol table, except names beginning with L and section name entries.
−S *addr file* ...	sets the location of the *files* listed. *Addresses* can be specified in hex, octal, or decimal.

sort sorts lines of all the named files together and writes the result on the standard output.

sort [**−bcdfimnortTu**]
[+*pos1* [−*pos2*]]
[**−o** *name*]
[**−T** *directory*] [*file*] ...

−b	ignores leading blanks (spaces and tabs) in field comparisons.
−c	checks that the input file is sorted according to the ordering rules; gives no output unless the file is out of sort.
−d	dictionary order: only letters, digits and blanks are significant in comparisons.
−f	folds uppercase letters onto lowercase.
−i	ignores characters outside the ASCII range 040-0176 in nonnumeric comparisons.
−m	merges only, the input files are already sorted.
−n	an initial numeric string, consisting of optional blanks, optional minus sign, and zero or more digits with optional decimal point, is sorted by arithmetic value.
−o *name*	the next argument names an output file other than the standard output.
−r	reverses the comparisons.
−t*x*	tab character separating fields is *x*.
−T *directory*	the next argument names a directory where temporary files are made.
−u	suppresses all but one in each set of equal lines.

spell, spellin, spellout collects words from the named document, and looks them up in a spelling list.

spell [**−bdvx**] ... [*file*] ...
spellin [*list*]
spellout [**−d**] *list*

−b	use British spelling.
−d	used with **spellout;** prints on the standard output those words on the hash list.
−v	all words not literally in the spelling list are printed, and plausible derivations from spelling list words are indicated.
−x	prints stems.

split splits the file into specified pieces.

split [−*n*] [*file* [*name*]]

−*n*	writes *file* in *n*-line pieces (default is 1000) or *n*-character pieces if number ends with a "c" (default is 10,000).

sprof displays profile data for segmented programs.

sprof [**−al** [*file*]]

−a	reports all symbols, rather than just external symbols.
−l	lists output by symbol value rather than decreasing percentage.

strings prints strings in object or other binary files.

strings [−] [−*number*] [**−o**] *file* ...

−	examines the entire file, even if in object format.
−*number*	*number* is the minimum string length instead of 4.
−o	gives each string's offset in octal.

strip removes symbols and relocation bits ordinarily attached to the output of the assembler and loader.

strip [**−h**] *file* ...

−h	strips the header and segment table.

stty sets the I/O options for a terminal; without arguments, reports settings of certain options.

stty [**−a**] [**−g**] [*options*]

−a	reports all option settings.
−g	reports current settings that can be used as arguments to other **stty** commands.

su substitutes user ID temporarily.

su [−] [*name* [*arg* ...]]

−	changes the environment to the user login.

sum calculates and prints a 16-bit checksum and the number of blocks in the file.

sum [**−r**] *file*

−r	uses the algorithm from the previous version of **sum** for computing the checksum.

tabs sets tab stops on user's terminal defined by tabspec; clears previous settings.

tabs [*tabspec*] [**+m***n*] [**−T***type*]

+m*n*	moves over *n* columns by making column *n*+1 the left margin; **−m** without *n* assumes a value of 10.
−T*type*	*type* is a terminal name listed in **term**(7).

tail prints the last 10 lines of a file. Copying can begin +*number* from the beginning or −*number* from the end of a file.

tail [±*number* [**bcl**]] [**−f**] [*file*]

b	*number* is blocks.
c	*number* is characters.
l	*number* is lines.
−f	"follow" option. If the input file is not a pipe, the program enters an endless loop. It can be used to monitor the growth of a file that is being written by some other process.

talk communicates with another user.

talk *user* [*ttyname*]

tar saves and restores files in an archive.

tar [**crtuxNbflmqvw**] [*file* ...]

c	creates a new tape: writes from beginning of archive.
r	appends files to existing files on the tape.
t	lists file names as they occur on the archive.
u	adds files to the archive if not there or changed since last put on the archive.
x	extracts files from the archive.
0,...,7	selects drive for tape mounting; default is 0.
b	the next argument is the blocking factor. The range is 1 to 20; default is 8.
f *file*	names archive or *file* instead of /dev/rct0.
l	prints error messages for unresolved links for files dumped.
m	does not restore the modification times.
q	does a quick extract. Retrieves the first occurrence of the file.

v	prints file name preceded by the function letter.
w	waits for user confirmation. If **y** is given, the action is performed.

tbl is a preprocessor for formatting tables for **nroff**(1) or **troff**(1).

tbl [*file*] ...

tee transcribes the standard input to the standard output and copies to files.

tee [**−a**] [**−i**] [*file*] ...

−a	appends output to *files* rather than overwriting them.
−i	ignores interrupts.

test evaluates files, strings, and numbers.

test *expr*

−r *file*	true if the file exists and is readable.
−w *file*	true if the file exists and is writable.
−x *file*	true if the file exists and is executable.
−f *file*	true if the file exists and is not a directory.
−d *file*	true if the file exists and is a directory.
−c *file*	true if the file exists and is a character special file.
−b *file*	true if the file exists and is a block special file.
−u *file*	true if the file exists and its set-user-ID bit is set.
−g *file*	true if the file exists and its set-group-ID bit is set.
−k *file*	true if the file exists and its sticky bit is set.
−s *file*	true if the file exists and has a size greater than zero.
−t [*fildes*]	true if file descriptor number *fildes* (1 by default) is associated with a terminal device.
−z *s1*	true if the length of string *s1* is zero.
−n *s1*	true if the length of the string *s1* is nonzero.
s1 = *s2*	true if the string *s1* and *s2* are equal.
s1 ! = *s2*	true if the strings *s1* and *s2* are not equal.
s1	true if *s1* is not the null string.
n1 **−eq** *n2*	true if *n1* and *n2* are algebraically equal.

time times a command.

time [**−v**] *command*

−v	verbose version.

timex times a command and generates a system activity report.

timex *command*

touch updates access and modification times of files.

touch [**−acm**] [*mmddhhmm*[*yy*]] *file* ...

−a	updates only the access time.
−c	does not create files that do not exist.
−m	updates only the modification time.

tr translates characters from one string to another string.

tr [**−cds**] [*string1* [*string2*]]

−c	complements the set of characters in *string1* with respect to characters whose ASCII codes are 001 through 377 octal.
−d	deletes all input characters in *string1*.
−s	squeezes strings of repeated output characters in *string2* to single characters.

troff formats text for typesetting.

troff [**−cikmnoqrsabfptTw**] ... [*file*] ...

−c*name*	prepends to input files the compacted macro files /usr/lib/macros/cmp.[nt].[dt].*name* and /usr/lib/macros/ucmp.[nt].*name*.
−i	reads standard input after the input files are exhausted.
−k*name*	compacts macros, places output in [dt].*name*.
−m*name*	prepends to input files the non-compacted macro file /usr/lib/tmac/tmac.*name*.
−n*N*	numbers first generated page *N*.
−o*list*	prints page numbers in range of *list*.
−q	invokes the simultaneous input/output mode of the **rd** request.
−r*aN*	sets register *a* (one-character) to *N*.
−s*N*	stops every *N* pages.
Troff only	
−a	sends a printable ASCII approximation of the results to the standard output.
−b	reports whether the phototypesetter is busy or available. No text processing is done.
−f	refrains from feeding paper and stopping phototypesetter at the end of the run.
−p*N*	prints all characters in point size *N*.
−t	directs output to the standard output instead of the phototypesetter.
−T*name*	uses font-width tables.
−w	waits until phototypesetter is available, if currently busy.

tsort produces a topological sort.

tsort [*file*]

tty prints the pathname of the user's terminal.

tty

uname prints the current system name of ZEUS on the standard output file.

uname [**−anrsv**]

−a	prints all available information.
−n	prints the nodename (name by which the system is known by to a communications network).
−r	prints the operating system release.
−s	prints the system name (default).
−v	prints the operating system version.

unget undoes a previous **get −e** of an SCCS file.

unget [**−n**] [**−r***SID*] [**−s**] *files*

−n	retains gotten file.
−r*SID*	identifies the delta which is no longer intended.
−s	suppresses the printout on the standard output of the intended delta's SID.

uniq reports repeated lines in a file.

uniq [**−cdu** [+*n*] [−*n*]] [*infile* [*outfile*]]

−c	outputs the number of times a line repeat occurred.
−d	writes one copy of repeated lines.
−u	lines not repeated in the original file are output.
n	specifies number of fields or characters to be ignored.

units converts quantities expressed in standard scales to equivalents in other scales.

units

unpack expands files (see **pack**).

unpack *files* ...

users lists the login names of the current users.

users

uucp is a ZEUS to ZEUS copy.

uucp [**−cCdefmn**] ... *source-file* ... *destination-file*

−c	uses the source file rather than copying the file to the spool directory (default).
−C	copies the source file to the spool directory.
−d	makes all necessary directories for the file copy (default).
−e*sys*	sends the **uucp** command to remote *sys* for execution.
−f	does not make intermediate directories for the file copy.
−m	sends the requestor mail when the copy is complete.
−n*user*	notifies *user* on the remote system that a file was sent.

uulog maintains a summary log of **uucp** and **uux**(1) transactions.

uulog [**-su**]...

-s*sys*	prints logging information about work involving *sys*.
-u*user*	prints logging information about work done for *user*.

uuname lists the **uucp** names of known systems. See **uucp**.

uuname

uupick (see **uuto** entry) queries for file disposition.

uupick [**-s***system*]

-s*system*	only searches for files sent by *system*.

uustat is a **uucp** status inquiry and job control.

uustat [**-cjkmosuvy**] ...

-c*hour*	removes status entries older than *hour*.
-j*all*	reports status of **uucp** requests.
-k*jobn*	kills the **uucp** request whose job number is *jobn*; must belong to the person issuing the **uustat** command or the super-user.
-m*mch*	reports accessibility status of machine *mch*.
-o*hour*	reports status of **uucp** requests older than *hour*.
-s*sys*	reports status of **uucp** requests which communicate with remote *sys*.
-u*user*	reports status of **uucp** requests issued by *user*.
-v	reports **uucp** status verbosely.
-y*hour*	reports status of **uucp** requests younger than hour.

uuto is the public ZEUS to ZEUS file copy which sends files.

uuto [*options*] *source-files destination*

-p	copies source-file into spool directory before transmission.
-m	sends mail to sender when the copy is complete.

uux is a ZEUS to ZEUS command execution.

uux [**-**] *command-string*

-	uses the standard input to the **uux** command as the standard input to the *command-string*.

val validates a SCCS file.

val - [**-m***name*]
[**-r***SID*]
[**-s**]
[**-y***type*] *files*

-	reads standard input line-by-line, processing arguments, until EOF.
-m*name*	compares the argument value *name* with the SCCS %M% keyword in *file*.
-r*SID*	specifies the delta to be validated.
-s	silences diagnostic messages generated on the standard output for errors detected while processing files on a given command line.
-y*type*	the argument value *type* is compared with the SCCS %Y% keyword in *file*.

vi is a screen-oriented (visual) editor based on **ex**(1).

vi [**-lrR**] [**-t** *tag*] [+ [*command*]] *files ...*

-l	LISP editing: the editing options Showmatch and Lisp are set.
-r	recovers files after an editor or system crash; the last saved version is retrieved.
-R	invokes a "read only" version of **vi**.
-t *tag*	positions the cursor at the definition of *tag* after **vi** is entered.
+ [*command*]	The editor begins by executing the command, *command*; if *command* is omitted, then the editor begins with the cursor positioned at the last line of the file.

vls visually lists files and directories.

vls [**-h**] *file ...*

-h	turns off highlighting of the cursored item.

vnews visually displays the news items.

vnews [**-h**]

-h	turns off highlighting of the cursored item.

vtzset sets up VTZ terminal function keys.

vtzset *file*

wc counts lines, words, and characters in file, or in the standard input if no name appears.

wc [**-clw**] [*file* ...]

-c	gives only character count.
-l	gives only line count.
-w	gives only word count.

what identifies SCCS files.

what *files*

whatis describes a command by giving the NAME line from the manual section.

whatis *name ...*

whereis locates source/binary and manual sections for specified files.

whereis [**−bms**] [**−u**] [**−BMS** *dir* ... **−f**] *file* ...

−b	searches for binaries.
−f	terminates the last such directory list and signals the start of file names.
−m	searches for manual sections.
−s	searches for sources.
−u	searches for unusual entries.
−[BMS]	changes or limits places **whereis** searches.

who lists the login name, terminal name, and login time for each current ZEUS user.

who [**−g**] [*****] [*who-file*] [**am I**]

−g	gives the group to which you belong.
*****	with three or more files in the **pwd**, gives the same information as **who** plus the logout time for terminals not logged in.
who-file	file used instead of /etc/utmp.
am I	gives login and terminal names, and login time.

whoami prints effective current user id.

whoami

whodo prints names and process status for current users.

whodo

whois accesses the user information data base.

whois [*name...*] [**−a** *name*] [**−m**]

name ...	prints specified users.
−a *name*	prints the *name* of user.
−m	adds new user to the data base.

write writes to another user.

write *user* [*ttyname*]

xargs constructs argument list(s) and executes commands.

xargs [*options*] [*command* [*initial-arguments*]]

−e*eofstr*	*eofstr* is the logical end-of-file string. **−e** without argument turns off logical EOF capability.
−i*replstr*	after executing, *command* is entered in *initial-arguments* for each occurrence of *replstr*.
−l*number*	*command* is executed for each non-empty *number* lines of arguments from standard input.
−n*number*	executes *command* using as many standard input arguments as possible, up to *number* arguments maximum.
−p	prompt mode: The user is asked whether to execute *command* for each invocation.

−s*size*	the maximum total size of each argument list is set to *size* characters.
−t	trace mode: The *command* and each constructed argument list are echoed to file descriptor 2 just prior to their execution.
−x	terminates if any argument list is greater than *size* characters.

xq examines or deletes requests from the line printer.

xq [**−d** *seqlist*]

xq [**−q** *que:dev* [**−s**]]

−d *seqlist*	*seqlist* is the list of sequence numbers to be removed from the queue.
−q *que:dev*	indicates which *queue* and *device* the **−s** option effects.
−s	stops printing.

xstr extracts strings from C programs to implement shared strings.

xstr [**−c**] [**−**] [*file*]

−c	extracts the strings from the C source in *file*, replacing string references by expressions of the form (&**xstr**[*number*]) for some number.
−	**xstr** reads from its standard input.

yacc converts a context-free grammar into a set of tables for a simple automaton that executes an **lalr**(1) parsing algorithm.

yacc [**−dv**] *grammar*

−d	generates the file y.tab.h with the define statements that associate the yacc-assigned token codes with the user-declared token names.
−v	prepares the file y.output containing a description of the parsing tables and a report on conflicts generated by ambiguities in the grammar.

zmenu drives menus created by **mkmenu**(1).

zmenu [*file*]

zmprint writes a menu, options list and helpfile to the standard output.

zmprint [*file*]

300, 300s handle special functions of DASI terminals.

300 [**−**dc,*1*,*t*] [**−***n*] [**+***12*]
300s [**−**dc,*1*,*t*] [**−***n*] [**+***12*]

−dc,*1*,*t*	controls delay factors.
−*n*	controls the size of half-line spacing.
+*12*	permits use of 12-pitch, 6 lines/inch text.

450 supports special functions and optimizes the use of the DASI 450 terminal, or any terminal that is functionally identical, such as the DIABLO 1620 or XEROX 1700.

450

ADMINISTRATOR COMMANDS

Administrator Commands, found in Section M of the ZEUS Reference Manual, are used by the system administrator/super-user to perform system maintenance. The super-user must have proper access permission and login as "zeus".

acctcms is a command summary from per-process accounting records.

acctcms [*options*] *files*

−a	prints output in ASCII.
−c	sorts by total CPU time, rather than total kcore-minutes.
−j	combines all commands invoked only once under "***other".
−n	sorts by number of command invocations.
−s	any file names encountered hereafter are already in internal summary format.

acctcom searches and prints process accounting files.

acctcom [[*options*] [*files*]]

−b	reads backwards, latest commands first.
−C *time*	shows only processes that exceed *time* indicating total CPU time.
−d *mm/dd*	following time arguments occur on the given month and day; looks at old files.
−e *time*	shows those processes that existed on or before *time*.
−f	prints the fork/exec flag and system exit status columns in the output.
−g *group*	shows *group* processes.
−h	shows "hogfactor," CPU time consumed by process execution (CPU time/elapsed time).
−H *factor*	shows processes that exceed *factor*.
−i	prints columns containing the I/O counts in the output.
−k	shows total kcore-minutes.
−l *line*	shows processes belonging to terminal /dev/*line*.
−m	shows mean core size (the default).
−n *pattern*	shows commands matching *pattern*.
−O *time*	shows processes with operating system CPU time exceeding *time*.
−r	shows CPU factor (*user-time/(system-time + user-time)*).

−s *time*	shows only those processes that existed on or after *time*, in the form *hr:min: sec*. The *:sec* or *:min:sec* may be omitted.
−t	shows separate system and user CPU times.
−u *user*	shows processes belonging to *user*.
−v	excludes column headings from the output.

acctconl creates an account summary for a sequence of login/logoff sessions.

acctconl [*options*]

acctcon2 converts a sequence of login sessions to total accounting records.

acctcon2 [*options*]

−l *file*	creates *file* summarizing line usage.
−o *file*	*file* is an overall record for the accounting period.
−p	prints input only, showing line name, login name, and time.
−t	lists the last time found in session record, assuring reasonable and repeatable numbers for non-current files.

acctdisk reads user ID, login name, and disk blocks and converts them to total accounting records.

acctdisk

acctdusg computes disk resource consumption.

acctdusg [**−p** *file*] [**−u** *file*] > *dtmp-file*

−p *file*	*file* is the name of the password file; not needed with /etc/passwd.
−u *file*	*file* consists of file names with no charges (a potential source for finding users trying to avoid disk charges).

acctmerg merges or adds total accounting files.

acctmerg [**−aiptuv**] [*file*]

−a	produces output in ASCII version of **tacct.**
−i	input files are in ASCII version of **tacct.**
−p	prints input with no processing.
−t	produces a single record that totals all input.
−u	summarizes by user ID, rather than user ID and name.
−v	produces output in verbose ASCII format, with more precise notation for floating point numbers.

accton turns process accounting off.

accton [*file*]

acctprcl summarizes user **acct** time data.

acctprcl [ctmp]

ctmp	lists login sessions sorted by user ID and login name.

acctprc2 summarizes and sorts records gathered by **acctprcl** and prints out.

acctwtmp writes a **wtmp**(5) record containing current time, name, and line to standard output.

acctwtmp [*name*[*line*]] >> /usr/adm/wtmp

adduser adds a new user to the system.

adduser

chargefee charges number dollars to login-name.

chargefee *login-name number*

chgrp changes group-ID of files to group.

chgrp *group file*

chmog, chog changes mode, owner and group of files to octal-mode, user, and group respectively.

chmog *octal-mode user group file*
chog *user group file*

chown changes the owner of the files to owner.

chown *owner file*

chroot changes root directory for a command.

chroot *newroot command*

ckpacct checks the size of /user/adm/pacct.

ckpacct [*blocks*]

clri clears i-nodes.

clri *filesystem i-number*

cron executes commands at specified dates and times.

cron

date prints and sets the date.

date [**−u**] [*mmddhhmm*[*yy*]] [+*format*]

−u	prints GMT time.

datem prompts for and sets the date and time.

datem

dcheck is a filesystem directory consistency check.

dcheck [**−i** *numbers*] [*filesystem*]

−i *numbers*	lists specified i-numbers which turn up in a directory. The number, the i-number of the directory, and the name of the entry are reported.

devnm identifies the special file associated with the mounted filesystem where name resides.

devnm [*name* ...]

df reports the number of free disk blocks.

df [**−f**] [**−t**] [*filesystems*]

−f	counts only blocks in the free list.
−t	reports total allocated block figures.

dodisk performs the disk accounting functions.

dodisk

down is a Shell script that takes the system down in an orderly manner.

down [**−bd**]

−b *time*	time in minutes between warning messages.
−d *time*	time in minutes before system shutdown.

dqueuer processes and removes print queue command requests from **nq.**

dqueuer [**−knr**]

−k	kills the **dqueuer**, although outstanding backends will finish their current file.
−n	(no execute) prints a summary of the configuration file.
−r	rereads /usr/spool/queuer/config to create the active configuration file.

dump is an incremental filesystem dump.

dump [*key* [*argument*] [*key* [*argument*]] *filesystem*]

a	if the dump is longer than one tape, **dump** will abort and print a message to the standard output.
b	allows specification of blocksize to make the tape.
f	places the dump on the next *argument* file instead of /dev/dumpdev.
h	using the next *argument*, writes a user comment onto the dump-tape header.
n	using the next *argument*, writes the filesystem name onto the dump-tape header.
s	the next *argument* is the length of the tape in feet.
u	if successful, write the date on '/etc/ddate' for each filesystem and each dump level.
0-9	dump level: dumps all files modified since the last date stored in the file '/etc/ddate.'

fsck audits and interactively repairs inconsistent conditions for the named filesystems.

fsck [[**−fnsSty**] *filesystem*]

−f	echoes output from the command into /tmp/fsck.err.
−n	assumes a "no" response to all questions.
−s	ignores the actual free list and constructs a new one by rewriting the super-block of the *filesystem.*
−S	conditionally reconstructs the free list if no discrepancies are discovered in the *filesystem.*
−t	the next argument names a scratch file for keeping tables.
−y	assumes a "yes" response to all questions.

fsdb patches a damaged filesystem after a crash.

fsdb *special-file* −

special-file	special device file.
−	disables error-checking routines.

fwtmp converts binary wtmp records to formatted ASCII records.

fwtmp [**−ic**]

−ic	denotes input is ASCII; writes output in binary form.

getty sets the modes of a terminal.

getty *name type delay*

halt is a Shell file to halt the system promptly.

halt

icheck is a filesystem storage consistency check.

icheck [**−s**] [**−b** *numbers*] *filesystem*

−b *numbers*	is a list of block *numbers*; if a named block turns up in a file, a diagnostic is produced.
−s	ignores actual free list and reconstructs a new one by rewriting the super-block of the *filesystem*.

icpcntrl starts and stops ICP protocols.

icpcntrl *icp#* [[**−start** [*protocol:ports*]] [**−stop** [*protocol:ports*]]

−start [*protocol:ports*]	starts an ICP 8/02 or indicated *protocol* on *ports* 0 to 8.
−stop [*protocol:ports*]	stops an ICP 8/02 or an indicated *protocol. ports* is same as above.

icpload loads and configures ICP 8/02 protocols.

icpload *icp#* [[**−k** *kernel*] [**−c** *ports:config*] *protocols*]

−c *ports:config*	configures *ports* which range from 0 to 8.
−k *kernel*	loads the segmented object module *kernel* as the kernel on the ICP 8/02.

init is invoked inside ZEUS as the last step in the boot procedure.

init [*state:id:flags:command*]

state	integer 1 to 9.
id	2 character identifier for active process.
flags	characters **t**, **K**, **c**, or **O**.
command	executable file name.

install installs a file (updated target file) in a specific place within a filesystem; commonly used in "makefiles" (see **make**(1)).

install [**−c** *dira*]
[**−f** *dirb*]
[**−i**]
[**−n** *dirc*]
[**−o**]
[**−s**]
file
[*dirx* ...]

−c *dira*	installs a new command in directory *dira file*; if *file* exists, **install** exits.
−f *dirb*	installs *file* in *dirb*, whether or not it already exists.
−i	ignores default directory list, searching only through the given directories (*dirx* ...).
−n *dirc*	if *file* is not found in any of the searched directories, it is put in the directory specified in *dirc*.
−o	if *file* is found, this saves the "found" file by copying it to OLD*file* in the directory in which it was found.
−s	suppresses messages other than error messages.
dirx	are directories searched before default directories.

labelit provides initial labels for unmounted disk or tape filesystems.

labelit *special* [*fsname volname* [**−n**]]

special	physical disk section or tape.
fsname volname	mounted name of filesystem being labeled.
−n	initial labeling of new tapes.

lastlogin shows the last date that a user logged in.

lastlogin

link exercises link system calls.

link *file1 file2*

lp services line printer spooler print requests.

/usr/lib/lp [*options*]

−B	generates banner pages stating the file could not be found.
−c *n*	prints the file *n* times.
−d *dest*	used in a destination portion of the banner page.
−f *filename*	filename, printed on the banner page.
−F *from*	indicates a 'from' string, for use on the banner page.
−s *spool time*	a string in **ctime**(3) format, indicating when the file was spooled to be printed. This value is printed on banner page.
−t *title*	a string used as a title on banner pages.
−T [*stty parameters*]	sets terminal parameters.

makekey generates an encryption key.

/usr/lib/makekey

makenewfs constructs and restores the filesystem.

makenewfs

makewhatis remakes the data base for the **whatis** and **apropos** commands.

makewhatis

mfs mounts all filesystems.

mfs

mkfs constructs a filesystem by writing on the special file with directions in the remainder of the command line, or to the prototype file, proto.

mkfs *special proto*

mkfs *special blocks interleave sectors*

mkmt makes special files for magnetic tape devices.

mkmt [**mt0**] [**mt1**] [**mt2**] [**mt3**]

mknod makes a directory entry and corresponding i-node for a special file.

mknod *file* [**c**] [**b**] *major minor*

mknod *file* **p**

c	character-type special file.
b	block-type special file.
p	creates fifo's (pipes).
major	major device number.
minor	minor device number.

mktape makes special files for cartridge tape devices.

mktape [**0**] [**1**] [**2**] [**3**]

monacct creates monthly accounting summary files; number shows which month/period.

monacct *number*

mount mounts filesystems.

mount [*special directory* [**−r**]]

special	device where removable filesystem is located.
directory	names an existing directory for root of new filesystem.
−r	mounts files as read-only; (must be used for physically write-protected and magnetic tape filesystems to prevent errors).

mvdir renames directories within a filesystem.

mvdir *oldname newname*

ncheck generates a path name vs. i-number list of files on specified filesystem.

ncheck [**−i** *numbers*] [**−as**] [*filesystem*]

inumbers	report only files with specified i-numbers.
−a	prints "." and ".." files.
−s	reports special files and set-user-ID mode.

nulladm creates file with 644 mode; owner is **adm.**

nulladm *file*

package extracts an optional package from a ZEUS release tape.

/etc/package

prctmp prints an accounting session record file.

prctmp

prdaily reports previous day's accounting.

prdaily

prtacct formats and prints any total accounting file.

prtacct *file* [*heading*]

pstat prints system facts.

pstat [**−afiptux**] [*suboptions*] [*file*]

−a	under **−p**, describes all process slots rather than just active ones.
−f	prints the open file table with descriptive headings.
−i	prints the i-node table with descriptive headings.
−p	prints process table for active processes with descriptive headings.
−t	prints the table for terminals (sio ports) with descriptive headings.
−u	prints information about a user process; the next argument is its address as given by **ps**(1).
−x	prints the text table with descriptive headings.
suboptions	are the descriptive table headings.

quot summarizes filesystem ownership.

quot [**−cfn**] *filesystem*

−c	prints three columns giving file size in blocks, number of files of that size, and cumulative total of blocks in that size or smaller file.
−f	prints count for number of files as well as space owned by each user.
−n	lists all files and their owners, with the pipeline: **ncheck** *filesystem* | **sort +0n** | **quot −n** *filesystem*

rc is a read command startup control script.
rc_csh is a read command C-shell multi-user startup script.

rc
rc_csh

reservrc installs or removes the **reserv**(1) utility for the system.

reservrc [**−i**] [**−r**]

−i	installs **reserv** into the system.
−r	removes **reserv** from the system.

restor reads magtapes dumped with the **dump** command.

restor *key* [*argument* ...]

key	must be present as one of the characters **lrRtx**; optionally combines with **dfvw.**
l	does everything **t** does, plus lists file names on the dump.
r or **R**	reads and loads the tape into the filesystem specified in *argument.*
t	prints the filesystem name, user comments made at dump time, date the tape was written and date the filesystem was dumped.
x	extracts files on the tape named in *argument*.
d	takes the next *argument* as the position (starting with 1) of the file on the tape with respect to the current position of the tape.
f	the next *argument* names the tape; default name is /dev/rct0.
v	only meaningful when used with **l**; gives a verbose listing of the file names including the owner group, permissions, and date of last modification.
w	prompts the user with the file name to be restored immediately before it is extracted.

rmuser removes a user from the system.

rmuser *name*

runacct performs the accumulation of connect, process, fee, and disk accounting on a daily basis; creates summaries of command usage.

runacct [*mmdd*] [*mmdd state*]

setlp displays and changes the line printer parameters.

setlp [**−dlsw**]

−d *devname*	the device file set or examined is *devname*.
−l *number*	sets the number of lines to the decimal *number*.
−s *number*	sets the starting column (numbered from 0) to the decimal *number*.
−w *number*	sets the number of columns to the decimal *number*.

setmnt creates the /etc/mnttab table needed for both the **mount** and **umount** commands.

setmnt

shut warns of system shutdown.

shut [**−bd**]

−b *time*	specified time in minutes between warnings.
−d *time*	specified time in minutes before shutdown.

shutacct should be invoked during a system shutdown to turn process accounting off and append a "reason" record to /usr/adm/wtmp.

shutacct [*reason*]

startup turns accounting on when system is brought up.

startup

str is the software trouble report input program.

str [**−l**]

−l	produces a line printer listing upon completion of the trouble report.

strprint prints a copy of all software trouble reports on the line printer.

strprint

sync updates the super-block.

sync

sysgen generates a ZEUS kernel.

sysgen [**−f** *file*] [**−d {11,21,31}**]

−f *file*	renames the resulting kernel *file* (zeus by default).
−d{11,21,31}	uses default answers to questions normally asked for Models 11, 21, and 31.

text prints spooler requests on text quality printers (see **lp**).

/usr/lib/text [*options*]

ttyconfig configures ports for terminal or modem line.

ttyconfig [**−m** *ports*] [**−t** *ports*]

−m *ports*	configures the user supplied list of *ports* for a modem.
−t *ports*	configures the user supplied list of *ports* for a terminal.

turnacct turns process accounting on or off.

turnacct [**on** \| **off** \| **switch**]

umfs unmounts all filesystems.

umfs

umount dismounts filesystems (see **mount**).

umount *special*

unlink exercises unlink system calls.

unlink *file*

update is a periodic buffer flush.

update

upkeep maintains a .contents file and its corresponding directory.

upkeep [**−dfilms**] [*directory*]

−d	reports the differences between .contents file and its corresponding *directory*.
−f	fast update of *directory*: changes the directory to match .contents file, produces a change report.
−i	initializes the .contents file with current specified *directory*.
−l	lists .contents file.
−m	modifies .contents file and *directory*.
−s	slow update, same action as fast update; displays a confirmation request before each change action.

uuclean provides **uucp** spool directory cleanup.

uuclean [*options*]

−d *directory*	cleans *directory* instead of the spool directory.
−m	sends mail to file owner after deletion.
−n *time*	deletes all files older than *time* if prefix test is satisfied.
−p *pre*	scans for files with *pre* as file prefix. **−p** deletes files older than specified time.

uusub monitors **uucp** network.

uusub [*options*]

−a*sys*	adds *sys* to the subnetwork.
−c*sys*	exercises the connection to the system *sys*.

—d*sys*	deletes *sys* from the subnetwork.
—f	flushes the connection statistics.
—l	reports statistics on connections.
—r	reports statistics on traffic amount.
—u*hr*	gathers traffic statistics over past *hr* hours.

wall writes to all users.

wall

wtmpfix changes time/data entries when using **fwtmp.**

wtmpfix [*files*]

xq examines or deletes requests from the line printer spooler.

xq *option* [*option*]

—o	used as the first option on the **xq** command line. Allows modification or deletion of requests belonging to another user. The user must be super-user or system group.
—b	backup device. Causes a device to backup its output one page (used when a printer has become fouled, or needs new paper loaded.)
—Dd	DOWN device. Sets a device to the DOWN state.
—Dq	DOWN queue. Sets a queue to DOWN state. No requests to this queue will be accepted while its status is DOWN.
—q *queue:device*	specifies queue and device in a request against a currently printing queue-member.
—r	restart. Restart at beginning on designated queue:device.
—s[kd]	quit printing. Printing on specified device stops immediately. With **k**, the file currently printing will be kept; with **d**, device status will be set to DOWN.
—Ud	UP device. Makes a device available for print selection.
—Uq	UP queue. Makes a queue available for print requests.

THE C SHELL

The C Shell acts as a command interpreter between the user and the operating system. It also interprets built-in control-flow commands (resembling those of the C language) for writing command scripts that execute like programs. This section lists the special characters, built-in commands, and their definitions. The notational conventions described in the "User Command" section apply here, with the addition of parentheses "( )" which indicate execution of a command in a subshell. Refer to the ZEUS Utilities Manual for a more extensive description of the C Shell.

Special Characters

|	pipeline
;	sequential command separator
( )	surrounds expressions and variable values
&	executes command as a background task
&&	executes next command only if previous command returned 0 status (completed)
| |	executes next command only if previous command returned non-zero value (failed)
\\ ... \\	immediate execution of enclosed command
'...'	takes enclosed characters literally
"..."	literal except for variable and command expansion
\\	negates special meaning of following character
#	comment line

Input-Output

<	redirects input
>	redirects standard output
>&	redirects standard and diagnostic output
>>	appends to a file
>!	overwrites a file that has noclobber set
>>!	appends to a file that has noclobber set

Filename Generation

/	separates the components of a pathname
?	matches a single character
*	matches a string of characters (including null)
[...]	matches a class of characters; a pair of characters separated by a – matches all characters lexically between the pair
~	used at the beginning of a filename to indicate home directory
{ }	specifies groups of arguments with common parts

History Manipulation

!!	repeats the last command
!*n*	repeats command number *n*
!*string*	repeats command starting with *string*
!$	produces the last argument of the previous command
!*	repeats all arguments except #0
^*x* **^***y*	executes last command substituting *x* for *y*
!!:*n*	repeats argument number *n* from last command

Built-In Commands

alias is an abbreviation for a longer command.

alias [*name* [*command-string*]]

alias	prints all aliases.
alias *name*	prints the alias for *name.*
alias *name command-string*	assigns *command-string* as the alias of name.

break resumes execution after the end of the nearest enclosing **foreach** or **while** loop.

break

breaksw breaks from a **switch**, resuming after the **endsw**.

breaksw

cd changes to a new working directory.

cd	returns to login directory.
cd *directory*	changes directory to *directory.*

continue continues execution of the nearest enclosing **while** or **foreach.**

continue

echo prints arguments to the standard output (terminal).

echo [**−n**] [*arg ...*]

−n	no newline is added to the output.

exit exits a shell.

exit
exit (*expression*)

false indicates failure status in a loop, returns nonzero exit status.

false

foreach provides flow control loop initiation.

foreach *name* (*list*)
command
end

getopt breaks up options in command lines for parsing; checks for legal options.

set -- getopt *optstring* **$***

--	delimits the end of the options.

gets can be used to read a string from the standard input.

gets [*default*]

glob prints strings on the terminal without spaces.

glob *wordlist*

history lists previous commands.

history
set history=*N*

if is a flow control branch statement.

if (*expression.1*) **then**
command.1
else if (*expression.2*) **then**
command.2
else *command.3*
endif

logout terminates a login session.

logout

nice adjusts the priority of the command by seven (default) or by specified number (number can be negative or positive).

nice [*number*] *command*

onintr regulates responses to interrupts in a shell script.

onintr [**−**] [*arg*] ...

onintr	restores default action of the shell on interrupts.
−	causes all interrupts to be ignored.
onintr *label*	causes the shell to execute a **goto** *label* when interrupt is received.

rehash re-makes the hash table of available commands.

rehash

repeat repeats a command.

repeat *count command*

set establishes or modifies a csh variable.

set [*name=value*]

setenv establishes or revises an environment variable.

setenv [*name value*]

source executes commands in a shell script in the current shell.

source *file*

switch is a flow control statement for decision making.

```
switch (string)
        case label1:
                command
        breaksw
        case label2:
                command
        breaksw
        default:
                command
        breaksw
endsw
```

time times a command.

> **time** [**−v**] *command* (csh internal and /bin/time)

−v	prints a verbose version of the command's output.

true provides truth values in a loop and returns zero exit status.

> **true**

umask sets file-creation mode mask.

> **umask** [*nnn*]

unset removes csh variables, as opposed to environment variables.

> **unset** *name*

unsetenv removes an environment variable from the list of set variables.

> **unsetenv** *name*

wait awaits completion of a process.

> **wait**

while is a flow control statement for loop initiation.

```
while (expression)
        command
end
```

LIBRARY ROUTINES

The Library Routines section summarizes the major ZEUS libraries. All routines are described fully in Section 3 of the ZEUS Reference Manual.

Standard I/O Routines

/usr/include/stdio.h is the include file for definitions.

clearerr resets error status on *stream*.

```
clearerr (stream)
FILE *stream;
```

ctermid generates a file name for terminal associated with the current process.

```
char *ctermid (s)
char *s;
```

cuserid generates the login name of the user.

```
char *cuserid (s)
char *s;
```

fclose flushes and closes *stream*.

```
fclose (stream)
FILE *stream;
```

fdopen returns a stream associated with the file descriptor *fildes*.

```
FILE *fdopen (fildes, type)
char *type;
```

feof returns nonzero if an EOF has been read in *stream*.

```
feof (stream)
FILE *stream;
```

ferror returns nonzero on a read/write error in *stream*.

```
ferror (stream)
FILE *stream;
```

fflush flushes *stream*.

```
fflush (stream)
FILE *stream;
```

fgetc is a function returning the next character from *stream*.

```
int fgetc (stream)
FILE *stream;
```

fgets reads *n*-1 characters from *stream* into the string *s*.

```
char *fgets (s, n, stream)
char *s;
FILE *stream;
int n;
```

fileno returns an integer file descriptor associated with *stream*.

```
fileno (stream)
FILE *stream;
```

fopen opens a file.

> **FILE *fopen** (*filename, type*)
> **char** **filename, *type;*

fprintf places formatted output on *stream*.

> **int fprintf** (*stream, format* [, *arg*] . . .)
> **FILE** **stream;*
> **char** **format;*

fputc is a function which puts the character c into *stream*.

> **fputc** (*c, stream*)
> **char** *c;*
> **FILE** **stream;*

fputs copies the null-terminated string *s* to *stream*.

> **fputs** (*s, stream*)
> **char** **s;*
> **FILE** **stream;*

fread reads data from *stream* into *ptr*.

> **fread** ((**char ***) *ptr, sizeof* (**ptr*)*, nitems, stream*)
> **FILE** **stream;*
> **int** *ptr, nitems;*

freopen substitutes *filename* in place of *stream*.

> **FILE *freopen** (*filename, type, stream*)
> **char** **filename, *type;*
> **FILE** **stream;*

fscanf reads formatted input from *stream*.

> **fscanf** (*stream, format* [, *pointer*] . . .)
> **FILE** **stream;*
> **char** **format;*

fseek sets the position of the next I/O operation on *stream*.

> **fseek** (*stream, offset, ptrname*)
> **FILE** **stream;*
> **long** *offset;*
> **int** *ptrname;*

ftell returns the current offset to the start of the file associated with *stream*.

> **long ftell** (*stream*)
> **FILE** **stream;*

fwrite writes data from *ptr* to *stream*.

> **fwrite** ((**char ***) *ptr, sizeof* (**ptr*)*, nitems, stream)*
> **FILE** **stream;*
> **int** *ptr, nitems;*

getc returns the next character from *stream*.

> **int getc** (*stream*)
> **FILE** **stream;*

getchar returns the next character from stdin.

> **int getchar** ()

gets reads the string *s* from stdin.

> **char *gets** (*s*)
> **char** **s;*

getw returns the next word from *stream*.

> **int getw** (*stream*)
> **FILE** **stream;*

pclose closes a *stream* originally opened with **popen**.

> **int pclose** (*stream*)
> **FILE** **stream;*

popen creates a stream pipe.

> **FILE *popen** (*command, type*)
> **char** **command, *type;*

printf places formatted output on stdout.

> **int printf** (*format* [*,arg*] . . .)
> **char** **format;*

putc outputs a character c to *stream*.

> **int putc** (*c, stream*)
> **char** *c;*
> **FILE** **stream;*

putchar outputs a character c to stdout.

> **putchar** (*c*)
> **char** c;

puts copies the null-terminated string *s* to stdout and appends a new line character.

> **puts** (*s*)
> **char** **s;*

putw outputs a word to *stream*.

> **putw** (*w, stream*)
> **FILE** **stream;*

rewind rewinds to the beginning of the file associated with *stream*.

> **rewind** (*stream*)
> **FILE** **stream;*

scanf reads formatted input from stdin.

> **scanf** (*format* [, *pointer*] . . .)
> **char** **format;*

setbuf causes I/O on *stream* to use the character array *buf* instead of an automatically allocated buffer.

> **setbuf** (*stream, buf*)
> **FILE** **stream;*
> **char** **buf;*

sprintf places formatted output followed by null character in the string *s*.

> **int sprintf** (*s, format* [, *arg*] . . .)
> **char** **s, *format;*

sscanf reads formatted input from the string *s*.

> **sscanf** (*s, format* [, *pointer*] . . .)
> **char** **s, *format;*

tmpfile creates a temporary file.

> **FILE *tmpfile** ()

tmpnam creates name for a temporary file.

> **char *tmpnam** (*s*)
> **char** **s;*

ungetc pushes a character back into an input *stream.*

> **ungetc** (*c, stream*)
> **FILE** **stream;*
> **char** *c;*

C Library Routines

a64l returns a long value that corresponds to a null-terminated base-64 representation *s.*

> **long a64l** (*s*)
> **char** **s ;*

abort generates an IOT signal.

> **abort ();**

abs returns the absolute value of integer *i.*

> **int abs** (*i*)
> **int** *i;*

asctime returns a pointer to an ASCII string representing the time defined in the structure *tm.*

> **#include <time.h>**
> **char *asctime** (*tm*)
> **struct** *tm *tm;*

atof returns a double representation of the string *nptr.*

> **double atof** (*nptr*)
> **char** **nptr;*

_atof returns a floating point representation of *nptr* in register f0.

> **_atof**(*nptr*)
> **char** **nptr;*

atofd returns a double representation of *nptr* in register rq4.

> **double atofd** (*nptr*)
> **char** **nptr;*

atofs returns a float representation of *nptr* in register rq4.

> **float atofs** (*nptr*)
> **char** **nptr;*

atoi returns an integer representation of the string *nptr.*

> **int atoi**(*nptr*)
> **char** **nptr;*

atol returns a long representation of the string *nptr.*

> **long atol** (*nptr*);
> **char** **nptr;*

bsearch finds the location of *key* in the table *base.*

> **char *bsearch** (*key, base, nel, width, compar*)
> **char** **key;*
> **char** **base;*
> **int** *nel, width;*
> **int** (**compar*) ();

calloc allocates space for an array of *nelem* elements of size *elsize.*

> **char *calloc** (*nelem, elsize*)
> **unsigned** *nelem, elsize;*

crypt returns a pointer to an encrypted password *key.*

> **char *crypt** (*key, salt*)
> **char** **key, *salt;*

ctime returns pointer to the converted ASCII string that represents the time pointed to by *clock.*

> **char *ctime** (*clock*)
> **long** **clock;*

ecvt converts a *value* to a null-terminated, rounded string of *ndigit* ASCII digits and returns a pointer to it.

> **char *ecvt** (*value, ndigit, decpt, sign*)
> **double** *value;*
> **int** *ndigit, *decpt, *sign;*

edata returns the first address about the initilized data region.

> **extern edata;**

encrypt returns an encrypted value if *edflag* is 0 or a decrypted value if *edflag* is 1. It uses the key set by **setkey.**

> **encrypt** (*block, edflag*)
> **char** **block;*
> **int** *edflag;*

end returns the first address above the uninitialized data region.

> **extern end;**

endgrent closes the input stream opened by a call to **getgrent** or its siblings.

> **#include <grp.h>**
> **int endgrent** ();

endpwent closes the password file opened by **getpwent** or its siblings.

> **#include <pwd.h>**
> **int endpwent** ();

etext returns the first address about the program text.

> **extern etext;**

fcvt is identical to **ecvt** except that the correct digit is rounded for Fortran F.format output with *ndigits* fraction digits.

> **char *fcvt** (*value, ndigit, decpt, sign*)
> **double** *value;*
> **int** *ndigit, *decpt, *sign;*

free makes available a block of memory, previously allocated by **malloc**, pointed to by *ptr.*

> **free** (*ptr*)
> **char** **ptr;*

frexp returns mantissa of double *value* as double quantity *x* of magnitude less than 1 and stores integer *n* such that *value* = x*2**n indirectly through *eptr*.

```
double frexp (value, eptr)
double value;
int *eptr;
```

gcvt is similar to **ecvt** except that it places the result in *buf* and returns a pointer to *buf*.

```
char *gcvt (value, ndigit, buf)
double value;
int ndigit;
char *buf;
```

getenv searches environment list for string of the form *name* = *value* and returns *value* if found, otherwise 0.

```
char *getenv (name)
char *name;
```

/usr/include/grp.h is the include file for **getgrent, getgrgid** and **getgrnam**.

getgrent reads single line from group file and returns pointer to structure containing the single line.

```
struct group *getgrent();
```

getgrgid reads group file until *gid* matches the numeric group ID field of the last line read and returns a pointer to the structure containing the last line read.

```
struct group *getgrgid (gid) int gid;
```

getgrnam reads until *name* matches the name field of the last line read and returns a pointer to the structure containing the last line read.

```
struct group *getgrnam (name) char *name;
```

getlogin returns pointer to login name as found in /etc/utmp.

```
char *getlogin ();
```

getopt puts in *optarg* a pointer to the next option letter in *argv* that matches a letter in *opstring* and puts in *optind* the *argv* index of the matched letter.

```
int getopt (argc, argv, optstring)
int argc;
char **argv;
char *optstring;
extern char *optarg;
extern int optind;
```

getpass reads password from /dev/tty or stdin after prompting with *prompt* and returns pointer to string.

```
char *getpass (prompt)
char *prompt;
```

getpw searches password file for *uid*, fills in *buf* with the corresponding line, and returns nonzero if *uid* not found.

```
getpw (uid, buf)
char *buf;
int uid;
```

/usr/include/pwd.h is the include file for **getpwent, getpwnam** and **getpwuid.**

getpwent reads next line in password file after it is opened.

```
struct passwd *getpwent ();
```

getpwnam returns a pointer to an object in the password file that matches *name*.

```
struct passwd *getpwnam (name)
char *name;
```

getpwuid returns a pointer to an object in the password file that matches *uid*.

```
struct passwd *getpwuid (uid)
int uid;
```

gsignal raises the signal, *sig*, previously defined by **ssignal.**

```
#include <signal.h>
int gsignal (sig)
int sig;
```

gtime is similar to **localtime** but returns a pointer to GMT.

```
#include <time.h>
struct tm *gmtime (clock)
long *clock;
```

index returns a pointer to the first occurrence of the character *c* in the string *s*.

```
char *index (s, c)
char *s, c;
```

The following routines have the form:

```
routine (c)
int (c)
```

/usr/include/ctype.h is the include file for these routines.

isalnum (c) returns nonzero if *c* is alphanumeric.

isalpha (c) returns nonzero if *c* is alphabetic.

isascii (c) returns nonzero if *c* is ASCII character.

isatty returns 1 if *fildes* is associated with terminal device, 0 otherwise.

```
isatty (fildes)
```

iscntrl (c) returns nonzero if *c* is a control character.

isdigit (c) returns nonzero if *c* is a digit.

isgraph (c) returns nonzero if *c* is a printing character, except 0 for space.

islower (c) returns nonzero if *c* is lowercase alphabetic.

isprint (c) returns nonzero if *c* is printable.

ispunct (c) returns nonzero if *c* is any punctuation character.

isspace (c) returns nonzero if *c* is a spacing character.

isupper (c) returns nonzero if *c* is uppercase alphabetic.

isxdigit (c) returns nonzero of *c* is a hex digit.

l3tol converts list of *n* three-byte integers packed into a string pointed to by *cp* into a list of long integers pointed to by *lp*.

```
l3tol (lp, cp, n)
long *lp;
char *cp;
int n;
```

l64a returns a pointer to base-64 representation that corresponds to a long value, *l*.

```
char *l64a (l)
long l ;
```

ldexp returns *value* *2**exp.

```
double ldexp (value, exp)
double value;
int exp;
```

localtime returns a pointer to the localtime defined in the structure *tm*.

```
#include <time.h>
struct tm *localtime (clock)
long *clock;
```

lsearch finds the location of *key* in the table *base*, adding *key* if not found.

```
char *lsearch (key, base, nelp, width, compar)
char *key;
char *base;
int *nelp;
int width;
int (*compar) ();
```

ltol3 reverses the conversion performed by **l3tol**.

```
ltol3 (cp, lp, n)
char *cp;
long *lp;
int n;
```

malloc returns a pointer to a block of *size* bytes beginning on word boundary.

```
char *malloc (size)
unsigned size;
```

mktemp replaces *template* by a unique file name and returns the address of template.

```
char *mktemp (template)
char *template;
```

modf returns the positive fractional part of *value* and stores the integer part indirectly through *iptr*.

```
double modf (value, iptr)
double value, *iptr;
```

monitor prepares an execution profile for an executable program created by the profiling option in **cc**(1).

```
monitor (lowpc, highpc, buffer, bufsize, nfunc)
int (*lowpc) (), (*highpc) ();
short buffer [];
int bufsiz, nfunc;
```

nlist gets the entries from name list.

```
#include <nlist.h>
nlist (filename, nl)
char *filename;
struct nlist nl [];
```

perror produces a short error message on stderr for the last error encountered during a system call.

```
perror (s)
char *s;
int deverr;
int sys_nerr;
char *sys_errlist [];
int errno;
```

putpwent is the reverse of **getpwent:** given a pointer to the password structure created by **getpwent**, it writes a line on stream *f* which matches the format of /etc/passwd.

```
#include <pwd.h>
int putpwent (p, f)
struct passwd *p;
FILE *f;
```

qsort implements a quick sort algorithm.

```
qsort (base, nel, width, compar)
char *base;
int (*compar) ();
int nel, width;
```

rand returns successive pseudo-random numbers in the range 0 to 32767.

```
rand ()
```

realloc changes the size of the block pointed to by *ptr* and returns a pointer to the block that is possibly moved.

```
char *realloc (ptr, size)
char *ptr;
unsigned size;
```

rindex returns a pointer to the last occurrence of character *c* in the string *s*.

```
char *rindex (s, c)
char *s;
char c;
```

setgrent rewinds the input stream opened by **getgrent** or its siblings.

```
#include <grp.h>
int setgrent ();
```

setkey returns a key used by the DES encryption algorithm.

```
setkey (key)
char *key;
```

setpwent rewinds the password file.

```
#include <pwd.h>
int setpwent ();
```

sleep suspends the current process for *seconds.*

sleep (*seconds*)
unsigned *seconds;*

srand initializes **rand** with *seed;* it reinitializes when *seed* = 1.

srand (*seed*)
int *seed;*

ssignal sets an action, *action,* for a signal type, *sig.*

#include <signal.h>
int (ssignal (*sig, action*)) ()
int *sig,* (**action*) ();

strcat appends the string *s2* to the string *s1.*

char *strcat (*s1, s2*)
char **s1, *s2;*

strchr returns a pointer to the first occurrence of character c in string **s.*

char *strchr (*s, c*)
char **s, c;*

strcmp compares string *s1* and string *s2* and returns an integer greater than, equal to, or less than 0 as *s1* is lexicographically greater than, equal to, or less than *s2.*

int strcmp (*s1, s2*)
char **s1, *s2;*

strcpy copies string *s2* to string *s1.*

char *strcpy (*s1, s2*)
char **s1, *s2;*

strcspn returns the length of initial segment of string *s1* consisting entirely of characters not in string *s2.*

int strcspn (*s1, s2*)
char **s1, *s2;*

strlen returns the number of non-null characters in string *s.*

int strlen (*s*)
char **s;*

strncat appends *n* characters from string *s2* to string *s1.*

char *strncat (*s1, s2, n*)
char **s1, *s2;*
int *n;*

strncmp compares at most *n* characters in *s1* and *s2* in the manner of **strcmp.**

int strncmp (*s1, s2, n*)
char **s1, *s2;*
int *n;*

strncpy copies *n* characters from *s2* to *s1.*

char *strncpy (*s1, s2, n*)
char **s1, *s2;*
int *n;*

strpbrk returns a pointer to the first occurrence of any character from string *s2,* in string *s1.*

char *strpbrk (*s1, s2*)
char **s1, s2;*

strrchr returns a pointer to last occurrence of character c in string *s.*

char *strrchr (*s, c*)
char **s, c;*

strspn returns the length of the initial segment of string *s1* which consists entirely of characters from string *s2.*

int strspn (*s1, s2*)
char **s1, *s2;*

strtok returns a pointer to the first character of the first token in string *s1* delimited by separators specified in string *s2.*

char **strtok* (*s1, s2*)
char **s1, *s2;*

swab copies *nbytes* pointed to by *from* to position pointed to by *to,* exchanging even/odd bytes.

swab (*from, to, nbytes*)
char **from, *to;*
int *nbytes;*

/usr/include/ctype.h is the include file for the following conversion routines.

These routines have the form:

routine (c)
int (c)

toascii (c) returns the ASCII equivalent of c, masking any non-ASCII standard characters.

tolower (c) returns a lowercase character corresponding to uppercase c.

_tolower (c) is similar to **tolower**, but requires an uppercase argument.

toupper (c) returns an uppercase character corresponding to lowercase c.

_toupper (c) is similar to **toupper**, but requires a lowercase argument.

ttyname returns a pointer to a pathname of the terminal device associated with the file descriptor *fildes.*

char *ttyname (*fildes*)

ttyslot returns index of current user's entry in /etc/utmp.

ttyslot ()

tzset sets the external variables *timezone* and *daylight* according to the environment variable TZ.

tzset ()

Math Library Routines

/usr/include/math.h is the include file for definitions.

acos returns the arc cosine of *x* in range 0 to pi.

```
double acos (x)
double x;
```

asin returns the arc sin of *x* in range −pi/2 to pi/2.

```
double asin (x)
double x;
```

atan returns the arc tangent of *x* in range −pi to pi.

```
double atan (x)
double x;
```

atan2 returns the arc tangent of *x/y* in range −pi to pi.

```
double atan2 (x, y)
double x, y;
```

cabs returns the sqrt (*x***x* + *y***y*).

```
double cabs (z)
struct { double x, y;} z;
```

ceil returns the smallest integral value not less than *x*.

```
double ceil (x)
double x;
```

cos returns the cosine of *x*.

```
double cos (x)
double x;
```

cosh returns the hyperbolic cosine of *x*.

```
double cosh (x)
double x;
```

exp returns the exponential function of *x*.

```
double exp (x)
double x;
```

fabs returns the absolute value of *x*.

```
double fabs (x)
double (x);
```

floor returns the largest integral value not greater than *x*.

```
double floor (x)
double x;
```

fmod returns the number *f* such that $x = iy + f$ for integer *i*, and $0 \leq f < y$.

```
double fmod (x, y)
double x, y;
```

gamma returns the GAMMA function of *x*.

```
double gamma (x)
double x;
```

hypot returns the sqrt

```
double hypot (x, y)
double x, y;
```

j0, j1, jn, y0, y1, yn are bessel functions.

```
double j0 (x)
double x;
double j1 (x)
double x;
double jn (n, x)
double x;
double y0 (x)
double x;
double y1 (x)
double x;
double yn (n, x)
double x;
```

log returns the natural logarithm of *x*.

```
double log (x)
double x;
```

log10 returns the base 10 logarithm of *x*.

```
double log10 (x)
double x;
```

pow returns *x* raised to power of *y*.

```
double pow (x, y)
double x, y;
```

sqrt returns the square root of *x*.

```
double sqrt (x)
double x;
```

sin returns the sine of *x*.

```
double sin (x)
double x;
```

sinh returns the hyperbolic sine of *x*.

```
double sinh (x)
double x;
```

tanh returns the hyperbolic tangent of *x*.

```
double tanh (x)
double x;
```

C-ISAM Library Routines

/usr/include/isam.h is the include file for definitions.

isaddindex adds index to a C-ISAM file.

```
isaddindex (isfd, keydesc)
int isfd;
struct keydesc *keydesc;
```

isaudit maintains an audit trail for a C-ISAM file.

```
isaudit (isfd, filename, mode)
int isfd;
char *filename;
int mode;
```

isbuild defines a C-ISAM file.

> **isbuild** (*filename, recordlength, keydesc, mode*)
> **char** **filename;*
> **int** *recordlength;*
> **struct** *keydesc *keydesc;*
> **int** *mode;*

isclose closes a C-ISAM file.

> **isclose** (*isfd*)
> **int** *isfd;*

isdelcurr deletes the current record.

> **isdelcurr** (*isfd*)
> **int** *isfd;*

isdelete deletes the specified record from a C-ISAM file.

> **isdelete** (*isfd, record*)
> **int** *isfd;*
> **char** *record* [];

isdelindex removes an index from a C-ISAM file.

> **isdelindex** (*isfd, keydesc*)
> **int** *isfd;*
> **struct** *keydesc *keydesc;*

iserase removes a C-ISAM file and any associated audit trail file.

> **iserase** (*filename*)
> **char** **filename;*

isindexinfo accesses a C-ISAM file's directory information.

> **isindexinfo** (*isfd, buffer, number*)
> **int** *isfd;*
> **int** *number;*
> **struct** *keydesc *buffer;*
> or
> **struct** *dictinfo *buffer;*

islock read-locks a C-ISAM file.

> **islock** (*isfd*)
> **int** *isfd;*

isopen opens a C-ISAM file for processing.

> **isopen** (*filename, mode*)
> **char** **filename;*
> **int** *mode;*

isread reads records from a C-ISAM file.

> **isread** (*isfd, buffer, mode*)
> **int** *isfd;*
> **char** *buffer* [];
> **int** *mode;*

isrelease unlocks records in a C-ISAM file.

> **isrelease** (*isfd*)
> **int** *isfd;*

isrename renames a C-ISAM file.

> **isrename** (*oldname, newname*)
> **char** **oldname;*
> **char** **newname;*

isrewcurr rewrites current record in a C-ISAM file.

> **isrewcurr** (*isfd, record*)
> **int** *isfd;*
> **char** *record* [];

isrewrite rewrites record in a C-ISAM file.

> **isrewrite** (*isfd, record*)
> **int** *isfd;*
> **char** *record* [];

isstart selects the current index and record within a C-ISAM file.

> **isstart** (*isfd, keydesc, length, record, mode*)
> **int** *isfd;*
> **struct** *keydesc *keydesc;*
> **int** *length;*
> **char** *record* [];
> **int** *mode;*

isuniqueid obtains a unique ID for a C-ISAM file.

> **isuniqueid** (*isfd, uniqueid*)
> **int** *isfd;*
> **long** **uniqueid;*

isunlock unlocks a C-ISAM file.

> **isunlock** (*isfd*)
> **int** *isfd;*

iswrite writes a record into a C-ISAM file.

> **iswrite** (*isfd, record*)
> **int** *isfd;*
> **char** *record* [];

lddbl, ldfloat, ldint, ldlong are C-ISAM load routines.

> **double lddbl** (*p*)
> **char** **p;*
>
> **float ldfloat** (*p*)
> **char** **p;*
>
> **int ldint** (*p*)
> **char** **p;*
>
> **long ldlong** (*p*)
> **char** **p;*

stdbl, stfloat, stint, stlong are C-ISAM store routines.

> **stdbl** (*d, p*)
> **double** *d;*
> **char** **p;*
>
> **stfloat** (*f, p*)
> **float** *f;*
> **char** **p;*
>
> **stint** (*i, p*)
> **int** *i;*
> **char** **p;*
>
> **stlong** (*l, p*)
> **long** *l;*
> **char** **p;*

Terminal Independent Routines

These functions use the terminal capability data base, /etc/termcap.

> **char** *PC;*
> **char** **BC;*
> **char** **UP;*
> **short** *ospeed;*

tgetent returns the /etc/termcap entry for terminal *name* in *bp.*

> **tgetent** (*bp, name*)
> **char** **bp, *name;*

tgetflag returns 1 if the capability *id* is present for the terminal; returns 0 otherwise.

> **tgetflag** (id)
> **char** **id;*

tgetnum returns the numeric value of capability *id.*

> **tgetnum** (*id*)
> **char** **id;*

tgetstr returns the string value of *id* in *area* and advances *area* pointer.

> **char *tgetstr** (*id, area*)
> **char** **id, **area;*

tgoto returns the cursor addressing string decoded from *cm* to go to column *destcol* in line *destline.*

> **char *tgoto** (*cm, destcol, destline*)
> **char** **cm;*
> **int** *destcol, destline;*

tputs returns the number of lines decoded in string *cp* in *affcnt.*

> **tputs** (*cp, affcnt, outc*)
> **register** *char *cp;*
> **int** *affcnt;*
> **int** (**outc*) ();

Programmer's Work Bench (PWB) Library Routines

any returns a 1 if character c is equal to any character in string *str.*

> **any** (c,*str*)
> **char** c, **str;*

anystr returns the offset (in *str1*) of the first character matched from *str2.*

> **anystr** (*str1,str2*)
> **char** **str1, *str2;*

balbrk finds the offset, in string *str*, of the first of the characters in the string *end* occurring outside of a balanced string.

> **balbrk** (*str,open,clos,end*)
> **char** **str, *open, *clos, *end;*

cat concatenates strings.

> **char *cat** (*dest,source1,source2,source3...sourcen,0*);
> **char *dest**, **source1, *source2, *source3, *sourcen;*

clean_up is a default cleanup routine to resolve external references.

> **clean_up** ()

curdir places the complete pathname of the current directory in string *path.*

> **curdir** (*path*)
> **char** **path;*

dname returns a pointer to the name of the directory that contains the file pointed to by *pathname.*

> **char *dname** (*pathname*)
> **char** **pathname;*

fatal is a general-purpose error handler.

> **fatal** (*msg*)
> **char** **msg;*

fdfopen provides file-descriptor interface to input/output routines.

> **FILE *fdfopen** (*fd,mode*)
> **int** *fd,mode;*

giveup changes directory to "/" if the argument is 0, sets IOT signal to system default (0), and calls **abort.**

> **giveup** (*dump*)
> **int** *dump;*

imatch returns 1 if string *prefix* is a prefix of string *str*; else returns 0.

> **imatch** (*prefix,str*)
> **char** **prefix, *str;*

index returns offset of first occurrence of *str2* in *str1* if string *str2* is substring of string *str1.*

> **index** (*str1,str2*)
> **char** **str1, *str2;*

lockit is a process semaphore implemented with files.

> **lockit** (*lockfile,count,pid*)
> **char** **lockfile*;
> **unsigned** *count,pid*;

move copies the first *n* characters from string *a* to string *b*.

> **char *move** (*a,b,n*)
> **char** **a*, **b*;
> **unsigned** *n*;

patoi converts an ASCII string to an integer.

> **patoi** (*str*)
> **char** **str*;

patol converts an ASCII string to a long integer.

> **long patol** (*str*)
> **char** **str*;

rename renames *oldname* to *newname.*

> **rename** (*oldname,newname*)
> **char** **oldname*, **newname*;

repeat copies the string *str* to the string *result repfac* times.

> **char *repeat** (*result,str,repfac*)
> **char** **result*, **str*;
> **unsigned** *repfac*;

satoi is similar to **patoi** except that it stores an integer value through an integer pointer *ip*, and returns a pointer to first non-numeric character encountered.

> **char *satoi** (*str,ip*)
> **char** **str*;
> **int** **ip*;

setsig sets signals.

> **setsig** ()

setsigl catches signals set by **setsig.**

> **setsigl** ()

sname returns a pointer to the "simple" name of the path-name *str.*

> **char *sname** (*str*)
> **char** **str*;

strend returns a pointer to the end (null byte) of the string *str.*

> **char *strend** (*str*)
> **char** **str*;

substr copies at most *len* characters from string *str* starting at *str*[*origin*] to string pointed to be *result.*

> **char *substr** (*str,result,origin,len*)
> **char** **str*, **result*;
> **int** *origin*;
> **unsigned** *len*;

trnslat copies string *str* to string *result* replacing any character found in string *old* with corresponding character from string *new.*

> **char *trnslat** (*str,old,new,result*)
> **char** **str*, **old*, **new*, **result*;

unlockit removes the *lockfile* created by **lockit.**

> **unlockit** (*lockfile,pid*)
> **char** **lockfile*;
> **unsigned** *pid*;

userdir returns user's login directory name.

> **char *userdir** (*uid*)
> **int** *uid*;

userexit is a default **userexit** routine to resolve external references.

> **userexit** (*code*)
> **int** *code*;

username returns user's login name.

> **char *username** (*uid*)
> **int** *uid*;

verify checks to see if string *str1* contains any characters not in string *str2.*

> **char *verify** (*str1,str2*)
> **char** **str1*, **str2*;

xalloc allocates a block of *size* bytes of memory.

> **xalloc** (*size*)
> **unsigned** *size*;

xcreat creates files with *name* and *mode,* and returns file descriptor.

> **xcreat** (*name,mode*)
> **char** **name*;
> **int** *mode*;

xfree frees a block of memory, previously allocated by **xalloc,** pointed to by *ptr.*

> **xfree** (*ptr*)
> **char** **ptr*;

xfreeall frees all memory allocated by **xalloc.**

> **xfreeall** ()

xlink creates a link to file *f1* named *f2.*

> **xlink** (*f1,f2*)
> **char** **f1*, **f2*;

xmsg generates an error message based on the external variable **errno.**

> **xmsg** (*file,func*)
> **char** **file*, **func*;

xopen opens a file *name* with *mode,* returning a file descriptor.

> **xopen** (*name, mode*)
> **char** **name*;
> **int** *mode*;

xpipe creates a pipe and returns a file descriptor *t.*

> **xpipe** (*t*)
> **int** **t*;

xunlink removes the entry for the file pointed to by *f*.

```
xunlink (f)
char *f;
```

xwrite writes *nbytes* bytes from address *buffer* to the file associated with *fildes*.

```
xwrite (fildes,buffer,nbytes)
int fildes;
char *buffer;
int nbytes;
```

zero sets to zero the area of memory *cnt* bytes long, starting at address *ptr*.

```
char *zero (ptr,cnt)
char *ptr;
int cnt;
```

zeropad replaces initial blanks with "0" characters in string *str*; *str* is returned.

```
char *zeropad (str)
char *str;
```

SYSTEM CALLS

System Calls, executed from within a program, provide direct entry into the ZEUS kernel. Error status will be returned where applicable. System Calls can be found in Section 2 of the ZEUS Reference Manual.

access determines accessibility of *file* according to *mode*, which is 4(read), 2(write), or 1(execute) or a combination thereof.

```
int access (file, mode)
char *name;
int mode;
```

acct turns accounting on or off by writing a record to *file* for each terminated process. An argument of null turns off accounting.

```
int acct (file)
char *file;
```

alarm schedules signal after specified *seconds*.

```
unsigned alarm (seconds)
unsigned seconds;
```

brk sets lowest unused location to a non-segmented *addr*.

```
int brk (addr)
char *addr;
```

chdir changes working directory to *dirname*.

```
int chdir (dirname)
char *dirname;
```

chmod changes *mode* of file *name*.

```
int chmod (name, mode)
char *name;
int mode;
```

chown changes *owner* and *group* of a file pointed to by *path*.

```
int chown (path, owner, group)
char *path;
int owner, group;
```

chroot changes root directory to *dirname*.

```
int chroot (dirname)
char *dirname;
```

close closes the file associated with *fildes*.

```
int close (fildes)
int fildes;
```

creat creates a new *file* with *mode*.

```
int creat (file, mode)
char *file;
int mode;
```

dup duplicates an open file descriptor, *fildes*.

```
int dup (fildes)
int fildes;
```

dup2 causes *fildes2* to be associated with the same files as *fildes.*

int dup2 (*fildes, fildes2*)
int *fildes, fildes2;*

execl executes a *file* when the number of arguments is known.

int execl (*file, arg0, arg1, ..., argn,* (char *) 0)
char **file, *arg0, *arg1, ..., *argn;*

execle is similar to **execl**, but a pointer to the shell environment is also passed.

int execle (*file, arg0, arg1, ..., argn,* (char *) 0, envp)
char **file, *arg0, *arg1, ..., *argn, *envp* [ ];

execlp is similar to **execl**, but it searches for *file* in a list of directories obtained from the environment.

int execlp (*file, arg0, arg1, ..., argn,* (char *) 0)
char **file, *arg0, *arg1, ..., *argn;*

execv executes a *file* when the number of arguments is unknown.

int execv (*file, argv*)
char **file, *argv* [ ];

execve is similar to **execv**, but a pointer to the shell environment is also passed.

int execve (*file, argv, envp*)
char **file, *argv* [ ], **envp* [ ];

execvp is similar to **execv**, but it searches for *file* in a list of directories obtained from the environment.

int execvp (*file, argv*)
char **file, *argv* [ ];

exit terminates a process and returns *status.*

exit (*status*)
int *status;*

_exit is similar to **exit**, but circumvents all cleanup.

_exit (*status*)
int *status;*

fcntl does file control for file associated with *fildes* according to *cmd* with *arg.*

#include <fcntl.h>
int fcntl (*fildes, cmd, arg*)
int *fildes, cmd, arg;*

fork spawns new processes.

int fork ()

fstat gets open file status by file descriptor.

#include <sys/types.h>
#include <sys/stat.h>
int fstat (*fildes, buf*)
int *fildes;*
struct stat **buf;*

ftime stores date and time in *tp.*

#include <sys/types.h>
#include <sys/timeb.h>
ftime (*tp*)
struct *timeb *tp;*

getegid gets effective group identity.

int getegid ()

geteuid gets effective user identity.

int geteuid ()

getgid gets group identity.

int getgid ()

getpgrp gets group process IDs.

int getpgrp ()

getpid gets process IDs.

int getpid ()

getppid gets parent process IDs.

int getppid ()

getuid gets user identity.

int getuid ()

ioctl performs input/output control on the special file associated with *fildes.*

#include <sys/ioctl.h>
int ioctl (*fildes, request, arg*)
int *fildes, request, arg;*

kill sends signal *sig* to a process *pid.*

int kill (*pid, sig*)
int *pid, sig;*

link links *file1* to *file2.*

int link (*file1, file2*)
char **file1, *file2;*

lkdata locks file associated with *fildes* according to *flag* for locked region *lkblk* against concurrent access.

#include <sys/lockblk.h>
long lkdata (*fildes, flag, lkblk*)
int *fildes, flag;*
struct *lockblk *lkblk;*

lock locks a process in primary memory. If *flag* is zero, process is unlocked.

int lock (*flag*)
int *flag;*

lseek moves the read/write pointer. If *whence* is 0, the pointer is set to *offset* bytes. If it is 1, the pointer is set to *offset* plus current location. If it is 2, the pointer is set to the size of the file plus *offset.*

long lseek (*fildes, offset, whence*)
long *offset;*
int *fildes, whence;*

mdmctl configures port for modem or terminal line.

mdmctl (*request, ismodem, flag*)
int *request;*
long **ismodem;*
int *flag;*

mknod makes a directory or a special *file* according to *mode.* Unless *mode* indicates a block or character special file, *dev* is ignored.

```
int mknod (file, mode, dev)
char *file;
int mode, dev;
```

mkseg makes a segment of *size* bytes with preferred segment number, *segno.*

```
char *mkseg (segno,size)
unsigned segno, size;
```

mount mounts a filesystem on the *special* file. *Directory* refers to the root of the filesystem. If *rwflag* = 0, the filesystem is writable.

```
int mount (special, directory, rwflag)
char *special, *directory;
int rwflag;
```

nice changes program priority by *incr.* Positive values get lower priority than normal.

```
int nice (incr)
int incr;
```

open opens *file* for reading (mode = 0) or writing (mode = 1) or both (mode = 2). File status flags are set to *oflag.*

```
#include <fcntl.h>
int open (file, oflag [, mode] )
char *file;
int oflag, mode;
```

pause stops until signal.

```
int pause ()
```

pipe creates an interprocess channel with file descriptor, *fildes.*

```
int pipe (fildes)
int fildes [2];
```

profil does an execution time profile for non-segmented programs.

```
int profil (buff, bufsiz, offset, scale)
char *buff;
int segno, bufsiz, offset, scale;
```

ptrace processes traces for non-segmented child processes (*pid*) according to *request.*

```
#include <signal.h> (if non-segmented parent)
#include <ssignal.h> (if segmented parent)
int ptrace (request, pid, addr, data)
int *addr;
int request, pid, data;
```

read reads from file into *buffer* of *nbytes* bytes.

```
int read (fildes, buffer, nbytes)
char *buffer;
int fildes, nbytes;
```

sbrk adds *incr* segmented bytes to program's non-segmented data space.

```
char *sbrk (incr)
int incr;
```

setgid sets group ID *gid.*

```
int setgid (gid)
int gid;
```

setpgrp sets process group ID.

```
int setpgrp ()
```

setuid sets user ID.

```
int setuid (uid)
int uid;
```

sgbrk changes the size of a data segment. *Addr* is a segmented address whose offset is equal to the new size of the segment.

```
char *sgbrk (addr)
char *addr;
```

sgstat returns highest segmented code address in *buffer.*

```
sgstat (buffer)
struct {
     char segno ;
     unsigned size ; }
*buffer ;
```

signal catches or ignores signals.

```
#include <signal.h> (non-segmented)
#include <ssignal.h> (segmented)
int (*signal (sig, func)) ()
int (*func) ();
int sig;
```

sprofil does an execution time profile for segmented programs.

```
int sprofil (segno, buff, bufsiz, offset, scale)
char *buff;
int segno, bufsiz, offset, scale;
```

sptrace processes traces for segmented child processes (*pid*) according to *request.*

```
#include <signal.h> (if non-segmented parent)
#include <ssignal.h> (if segmented parent)
int sptrace (request, pid, addr, data)
int *addr;
int request, pid, data;
```

ssgbrk changes the size of data segment *segno* by increment *incr.*

```
char *ssgbrk (segno, incr)
unsigned segno, incr;
```

stat returns *file* status in *buf.*

```
#include <sys/types.h>
#include <sys/stat.h>
int stat (file, buf)
char *file;
struct stat *buf;
```

stime sets the time.

```
int stime (tp)
long *tp;
```

sync updates super-block.

```
sync ()
```

time returns the date and time.

```
long time ((long *) 0)
long time (tloc)
long *tloc;
```

times returns time and accounting information for the current process.

```
long times (buffer)
struct tbuffer *buffer;
```

ulimit gets and sets user limits.

```
long ulimit (cmd, newlimit)
int cmd;
long newlimit;
```

umask sets file creation mode masks.

```
int umask (complmode)
int complmode;
```

umount removes a filesystem from special file, *file*.

```
int umount (special)
char *special;
```

uname gets name of current Zilog system.

```
#include <sys/utsname.h>
int uname (name)
struct utsname *name;
```

unlink removes the directory entry for the file pointed to by *file*.

```
int unlink (file)
char *file;
```

unlk unlocks data against concurrent access.

```
#include <sys/lockblk.h>
long unlk (fildes, flag, lkblk)
int fildes, flag;
struct lockblk *lkblk;
```

ustat gives filesystem statistics.

```
#include <sys/types.h>
#include <ustat.h>
int ustat (dev, buf)
int dev;
struct ustat *buf;
```

utime sets file times.

```
#include <sys/types.h>
int utime (file, times)
char *file;
struct utimbuf *times;
```

wait waits for processes to terminate.

```
int wait (status)
int *status;
int wait ((int *) 0)
```

write writes on a file from *buffer* of *nbytes* bytes.

```
int write (fildes, buffer, nbytes)
char *buffer;
int fildes, nbytes;
```

TEXT PROCESSING

The Text Processing section summarizes four text manipulation tools. Vi is used to edit text and input formatting macros; MM is a set of macros; NROFF/TROFF shapes the text defined by the macros for printer or typesetter output. All sections are explained more fully in the ZEUS Utilities Manual.

VI

This section contains a summary of commands for manipulating the cursor and text when using the Visual Screen Editor, Vi.

Entering/Leaving Vi

%vi [*options*] *file*	edit *file* at top, options described in User Commands under Vi.
ZZ	exit from Vi, saving changes

Vi Modes

command	normal or initial mode, return here after completing command.
insert	entered with **aAiIoOcCsSrR**, terminate with **ESC**.

Insert and Replace Mode Selection

a	append after cursor
i	insert before cursor
A	append at end of line
I	insert before first non-blank
o	insert newline on line below cursor
O	insert newline at present cursor position, push current line down
r*x*	overwrite single character and return to command mode
R	overwrite characters while advancing cursor

Interrupting, Cancelling

ESC	end insert or incomplete command
^?	(delete or rubout) interrupts
^L	reprint screen if ^? scrambles it

File Manipulation

:w	write back changes
:wq	write and quit
:q	quit
:q!	quit; discard changes
:e *name*	edit file *name*
:e!	re-edit, discard changes
:e *#*	edit alternate file (also CTRL-↑)
:w *name*	write file *name*
:w! *name*	overwrite file *name*
:! *cmd*	run command *cmd*, then return
:n	edit next file in argument list
:f	show current file and line number (also CTRL-g)
:sh	escape to the type of shell defined in your environment
:csh	escape to C shell

Screen Positioning within a File

CTRL-f	forward screenful
CTRL-b	backward screenful
CTRL-d	scroll down half-screen
CTRL-u	scroll up half-screen
*n***G**	go to line *n* (end default)
/<*string*>	next line matching <*string*>
?<*string*>	previous line matchng <*string*>
n	repeat last / or **?**
N	reverse last / or **?**
/<*string*>/+*n*	*n'th* line after <*string*>
?<*string*>?−*n*	*n*'th line before <*string*>
]]	next section/function
[[	previous section/function
%	find matching parenthesis or brace

Cursor Positioning within a Screen

H	home window line
L	last window line
M	middle window line
+	next line, at first non-white
−	previous line, at first non-white
RETURN	same as carriage return; moves cursor to beginning of next line
j	next line, same column
k	previous line, same column

Cursor Positioning within a Line

↑	first non-white
0	beginning of line
$	end of line
l or −>	forward
h or <−	backwards
CTRL-H	same as <−
space	same as −>
f*x*	find *x* forward
F*x*	find *x* backward
t*x*	move-up to *x*
T*x*	back-up to *x*
;	repeat last f, F, t, or T
,	inverse of ;
|	to specified column

Cursor Positioning by Words, Sentences, Paragraphs

w	word forward
b	back word
e	end of word
)	beginning of next sentence
}	beginning of next paragraph
(	beginning of previous sentence
{	beginning of previous paragraph
W	blank delimited word
B	back **W**
E	to end of **W**

Corrections During Insert Mode

CTRL-H	erase last character
CTRL-W	erases last word
erase	your erase character; same as CTRL-h
kill	your kill character; erase input this line
\	escapes CTRL-h; your erase and kill
ESC	ends insertions, back to command mode
CTRL-?	interrupt, terminates insert
CTRL-D	backtab over *autoindent*
↑CTRL-D	kill *autoindent*, for one line only
0CTRL-D	kills all *autoindent*
CTRL-V	quote non-printing character

Operators (Double to Affect Lines)

d	delete
c	change
<	left shift
>	right shift
!	filter through command
=	indent for LISP
y	yank lines to buffer

Miscellaneous Operations

C	change rest of line
D	delete rest of line
s	substitute characters
S	substitute lines
J	join lines
x	delete character at cursor
X	delete character before cursor
Y	yank lines

Marking and Returning

``	return to previous position in text
''	cursor moves to first non-white character on the line at the previous position
m*x*	mark position with letter *x*
`*x*	to mark *x* at position within line
'*x*	to mark *x* at first non-white character in line

Yank and Put

p	put back line(s) after current line
P	put back line(s) before current line
"*x*p	put from buffer *x*
"*x*d	delete into buffer *x*
"*x*y	yank to buffer *x*

Undo, Redo, Retrieve

u	undo last change
U	restore current line
.	repeat last change
"*n*p	retrieve *n*'th last delete

Display Indications

Last line	states error messages, echoing input to :, /, ?, and !; feed-back about I/O and large changes.
@lines	on screen only (on dumb terminals), a place holder for deleted lines; not in file
~lines	lines past end of file
CTRL-x	control characters, CTRL-? is delete
tabs	expand to spaces, cursor at last

Simple Commands

dw	delete a word
de	delete a word, leave punctuation
dd	delete a line
3dd	delete 3 lines
i*text*ESC	insert text *text*
cw*new*ESC	change word to *new*
easESC	pluralize word
xp	transpose characters

NROFF/TROFF

Nroff/Troff commands interspersed in text are non-printing requests allowing documents to be formatted easily. The commands are entered on a separate line and usually begin with a period. Many commands accept parameter values and/or assume initial values (defaults). Nroff outputs to a printer, Troff to a typesetter.

General Explanation

;	Separates nroff and troff values, respectively.
†	No effect in nroff.
'	Using ' as a control character (instead of .) suppresses the break function.
N	Is a decimal number or a decimal-fraction that will be rounded to an integer number of basic units when stored as a parameter.
|N	Is the distance to place N from current place.
±N	Parameter may be N, +N (increment), or −N (decrement).

Fonts (F) = Times Roman (R), Times Italic (I), Times Bold (B), Special Math (S), and Previous (P) on physical typesetter positions 1-4.

Font and Character Size Control

Request Form	Initial Value	If No Argument	Notes*	Explanation
.ps ± N	10 point	previous	E	Point size; also an in-line request as \ s ± N.†
.ss N	12/36 em	ignored	E	Space-character size set to N/36 em.†
.cs F N *M*	off	—	P	Constant character spacing (width) mode for (font F).† Character width is N/36 ems. M defines em as M/points.
.bd F N	off	—	P	Embolden font F; double-print separated by N – 1 units.†
.bd S F N	off	—	P	Embolden Special Font when current font is F.†
.ft F	Roman	previous	E	Change to font F = x, xx, or 1 – 4; also \ fx, \ f(xx, \ fN.†
.fp N F	R,I,B,S	ignored	—	Font named F mounted on physical position N = (1-4).†

Page Control

Request Form	Initial Value	If No Argument	Notes	Explanation
.pl ±N	11 in	11 in	v	Page length.
.bp ±N	N=1	—	B,v	Eject current page; next page number N.
.pn ±N	N=1	ignored	—	Next page number N.
.po ±N	0;26/27 in	previous	v	Page offset. Left margin ±N.
.ne N	—	N=1V	D,v	Need N vertical space (V = vertical spacing).
.mk *R*	none	internal	D	Mark current vertical place in register *R*.
.rt ±N	none	internal	D, v	Return (upward only) to marked vertical place.

*Explanations of Note characters are found at the end of this section.

Text Filling, Adjusting, and Centering

Request Form	Initial Value	If No Argument	Notes	Explanation
.br	—	—	B	Break.
.fi	fill	—	B,E	Fill output lines.
.nf	no fill	—	B,E	No filling or adjusting output lines.
.ad l,c,r,b	adj, both	adjust	E	With fill, adjust output lines. l = left margin, r = right margin, c = center or a number from j register, and b = both.
.na	adjust	—	E	No output line adjusting.
.ce N	off	N=1	B,E	Center following N input text lines.

Spacing

Request Form	Initial Value	If No Argument	Notes	Explanation
.vs N	1/6 in; 12 pts	previous	E,p	Vertical base line spacing (V).
.ls N	N=1	previous	E	Output N – 1 Vs after each text output line.
.sp N	–	N=1V	B,v	Space vertical distance N in either direction.
.sv N	—	N=1V	B,v	Save vertical distance N.
.os	—	—	—	Output saved vertical distance.
.ns	space	—	D	Turn no-space mode on.
.rs	—	—	D	Restore spacing; turn no-space mode off.
.ll ±N	6.5 in	previous	E,m	Line length.
.in ±N	N=0	previous	B,E,m	Indent.
.ti ±N	—	ignored	B,E,m	Temporary indent.

Tabs, Leaders, and Fields

Request Form	Initial Value	If No Argument	Notes	Explanation
.ta N*t* ...	0.8;0.5 in	none	E,m	Tab settings; left type, unless *t* = R (right), C (centered).
.tc *c*	none	none	E	Tab repetition character.
.lc *c*		none	E	Leader repetition character.
.fc *a b*	off	off	—	Set field delimiter *a* and pad character *b*.

Macros, Strings, Diversion, and Position Traps

Request Form	Initial Value	If No Argument	Notes	Explanation
.de *xx yy*	—	.*yy*=..	—	Define or redefine macro *xx*; end at call of *yy*.
.am *xx yy*	—	.*yy*=..	—	Append to a macro.
.ds *xx string*	—	ignored	—	Define a string *xx* containing *string*.
.as *xx string*	—	ignored	—	Append *string* to string *xx*.
.rm *xx*	—	ignored	—	Remove request, macro, or string.
.rn *xx yy*	—	ignored	—	Rename request, macro, or string *xx* to *yy*.
.di *xx*	—	end	D	Divert output to macro *xx*.
.da *xx*	—	end	D	Divert and append to *xx*.
.wh N *xx*	—	—	v	Set location trap; negative is with respect to page bottom.
.ch *xx* N	—	—	v	Change trap location.
.dt N *xx*	—	off	D,v	Set a diversion trap.
.it N *xx*	—	off	E	Set an input-line count trap.
.em *xx*	none	none	—	End-macro is *xx*.

Number Registers

Request Form	Initial Value	If No Argument	Notes	Explanation
.nr *R*±N *M*	—	—	u	Define and set number register *R* with value ±N; auto-increment by *M*.
.af *R c*	arabic	—	—	Assign numbering sequence format to register *R* (c=1, i, I, a, A).
.rr *R*	—	—	—	Remove register *R*.

Three Part Titles

Request Form	Initial Value	If No Argument	Notes	Explanation
.tl 'left'center' right'	—	—	—	Three-part title placement.
.pc *c*	%	off	—	Page number character.
.lt ±N	6.5 in	previous	E,m	Length of title.

Input and Output Conventions and Character Translations

Request Form	Initial Value	If No Argument	Notes	Explanation
.ec c	\	\	—	Set escape character.
.eo	on	—	—	Turn off escape character mechanism.
.lg N	—;on	on	—	Ligature mode on if N>0.
.ul N	off	N=1	E	Underline; (italicize in troff) N input lines.
.cu N	off	N=1	E	Continuous underline in nroff; like .ul in troff.
.uf F	Italic	Italic	—	Underline font set to F (to be switched to by .ul).
.cc c	.	.	E	Set control character to *c*.
.c2 c	'	'	E	Set nobreak control character to *c*.
.tr *abcd*....	none	—	O	Translate *a* to *b*, etc. on output.

Hyphenation

Request Form	Initial Value	If No Argument	Notes	Explanation
.nh	hyphenate	—	E	No hyphenation.
.hy N	hyphenate	hyphenate	E	Hyphenate if N≥1.
.hc c	\%	\%	E	Hyphenation indicator character *c*.
.hw *word1*...	—	ignored	—	Exception words.

Output Line Numbering

Request Form	Initial Value	If No Argument	Notes	Explanation
.nm ±N *M S I*	—	off	E	Number-mode on or off, set parameters. ±N numbers next line output. *M* is multiple of N. *S* is text-number separation. *I* is a line number indent.
.nn N	—	N=1	E	Do not number next N lines.

Conditional Acceptance of Input

Request Form	Initial Value	If No Argument	Notes	Explanation
.if c *anything*	—	—	—	If condition c true, accept *anything* as input, for multi-line use \{*anything*\}.
.if !c *anything*	—	—	—	If condition c false, accept *anything*.
.if N *anything*	—	—	u	If expression N>0, accept *anything*.
.if !N *anything*	—	—	u	If expression N≤0, accept *anything*.
.if *string1 string2 anything*	—	—	—	If *string1* identical to *string2*, accept *anything*.
.if ! *string1 string2 anything*	—	—	—	If *string1* not identical to *string2*, accept *anything*.
.ie c *anything*	—	—	u	If portion of if-else; all above forms (like if).
.el *anything*	—	—	—	Else portion of if-else.

Environment Switching

Request Form	Initial Value	If No Argument	Notes	Explanation
.ev N	N=0	previous	—	Environment switched to 0≤N≤2 (pushed down).

Insertions from the Standard Input

Request Form	Initial Value	If No Argument	Notes	Explanation
.ab *text*	—	—	—	Prints *text* on message output and terminates.
.rd *prompt*	—	prompt= BEL	—	Read insertion.
.ex	—	—	—	Exit from nroff/troff.

Input/Output File Switching

Request Form	Initial Value	If No Argument	Notes	Explanation
.so *name*	—	—	—	Interpolates *name* at point of **.so** request (push down).
.nx *filename*		EOF	—	Next *filename*.
.pi *program*		—	—	Pipe output to *program* (nroff only).

Miscellaneous

Request Form	Initial Value	If No Argument	Notes	Explanation
.mc c N	—	off	E,m	Set margin character c and separation N.
.tm *string*	—	newline	—	Print *string* on terminal (ZEUS standard message output).
.ig *yy*	—	.yy=..	—	Ignore input until call of *yy*.
.pm *t*	—	all	—	Print macro names and sizes; *t* is the total of sizes.
.fl	—	—	B	Flush output buffer. With **−k**, compact current state of nroff/troff.

Notes

B	Request normally causes a break.
D	Mode or relevant parameters associated with current diversion level.
E	Relevant parameters are a part of the current environment.
O	Must stay in effect until logical output.
P	Mode must be currently or again in effect at the time of physical output.
v,p,m,u	Default scale indicator; if not specified, scale indicators are ignored.

Memorandum Macros (MM)

The MM command macros and arguments can be inserted within text, and, when processed with the Nroff or Troff text processors, produce output formatted to specification.

.1C—one-column processing

.2C—two-column processing

.AE—abstract end

.AF—alternate format of "Subject/Date/From" block

  .AF [*company name*]

.AL—automatically incremented list start

  .AL [*type*] [*text-indent*]
  type specifies numerical, alphabetical or mixed sequential listing.

.AS—abstract start

  .AS [*arg*] [*indent*]

.AT—author's title

  .AT [*title*] ...

.AU—author information

  .AU *name* [*initials*] [*loc*] [*dept*] [*ext*] [*room*] [*arg*] [*arg*] [*arg*]

.AV—approval signature line

  .AV [*name*]

.B—bold

  .B [*bold-arg*] [*previous-font-arg*] [*bold*] [*prev*] [*bold*] [*prev*]

.BE—bottom block end

.BI—bold/italic

  .BI [*bold-arg*] [*italic-arg*] [*bold*] [*italic*] [*bold*] [*italic*]

.BL—bullet list start

  .BL [*text-indent*]

.BR—bold/roman

  .BR [*bold-arg*] [*roman-arg*] [*bold*] [*roman*] [*bold*] [*roman*]

.BS—bottom block start

.CS—cover sheet

  .CS [*pages*] [*other*] [*total*] [*figs*] [*tbls*] [*refs*]

.DE—display end

.DF—display floating start

  .DF [*format*] [*fill*] [*right-indent*]

.DL—dash list start

  .DL [*text-indent*]

*Macros marked with an asterisk are not, in general, called (invoked) directly by the user. They are "user exits" defined by the user and called by the MM macros from inside header, footer, or other macros.

.DS—display static start

  .DS [*format*] [*fill*] [*right-indent*]

.EC—equation caption

  .EC [*title*] [*override*] [*flag*]

.EF—even-page footer

  .EF [*arg*]

.EH—even-page header

  .EH [*arg*]

.EN—end equation display

.EQ—equation display start

  .EQ [*label*]

.EX—exhibit caption

  .EX [*title*] [*override*] [*flag*]

.FC—formal closing

  .FC [*closing*]

.FD—footnote default format

  .FD [*arg*]

.FE—footnote end

.FG—figure title

  .FG [*title*] [*override*] [*flag*]

.FS—footnote start

  .FS [*label*]

.H—heading,numbered

  .H *level* [*heading-text*] [*heading-suffix*]

.HC—hyphenation character

  .HC [*hyphenation-indicator*]

.HM—heading mark style (arabic or roman numerals, or letters)

  .HM [*arg1*] ... [*arg7*]

.HU—heading,unnumbered

  .HU *heading-text*

***.HX**—heading user exit X (before printing heading)

  .HX *dlevel rlevel heading-text*

***.HY**—heading user exit Y (before printing heading)

  .HY *dlevel rlevel heading-text*

***.HZ**—heading user exit Z (after printing heading)

  .HZ *dlevel rlevel heading-text*

.I—italic (underline in the **nroff** formatter)

  .I [*italic-arg*] [*previous-font-arg*] [*italic*] [*prev*] [*italic*] [*prev*]

.IB—italic/bold

  .IB [*italic-arg*] [*bold-arg*] [*italic*] [*bold*] [*italic*] [*bold*]

.IR—italic/roman

  .IR [*italic-arg*] [*roman-arg*] [*italic*] [*roman*] [*italic*] [*roman*]

.LB—list begin

.LB *text-indent mark-indent pad type* [*mark*] [*LI-space*] [*LB-space*]

.LC—list-status clear

.LC [*list-level*]

.LE—list end

.LI—list item

.LI [*mark*]

.ML—marked list start

.ML *mark* [*text-indent*]

.MT—memorandum type

.MT [*type*] [*addressee*] or **.MT**

.ND—new date

.ND *new-date*

.NE—notation end

.NS—notation start

.NS [*arg*]

.nP—double-line indented paragraphs

.OF—odd-page footer

.OF [*arg*]

.OH—odd-page header

.OH [*arg*]

.OK—other keywords for the Technical Memorandum cover sheet

.OK [*keyword*] ...

.OP—odd page

.P—paragraph

.P [*type*]

.PF—page footer

.PF [*arg*]

.PH—page header

.PH [*arg*]

.PM—proprietary marking

.PM [*code*]

*****.PX**—page-header user exit

.R—return to regular (roman) font

.RB—roman/bold

.RB [*roman-arg*] [*bold-arg*] [*roman*] [*bold*] [*roman*] [*bold*]

.RD—read insertion from terminal

.RD [*prompt*] [*diversion*] [*string*]

.RF—reference end

.RI—roman/italic

.RI [*roman-arg*] [*italic-arg*] [*roman*] [*italic*] [*roman*] [*italic*]

.RL—reference list start

.RL [*text-indent*]

.RP—produce reference page

.RP [*arg*] [*arg*]

.RS—reference start

.RS [*string-name*]

.S—set **troff** formatter point size and vertical spacing

.S [*size*] [*spacing*]

.SA—set adjustment (right-margin justification) default

.SA [*arg*]

.SG—signature line

.SG [*arg*]

.SK—skip pages

.SK [*pages*]

.SM—make a string smaller

.SM *string1* [*string2*] [*string3*]

.SP—space vertically

.SP [*lines*]

.TB—table title

.TB [*title*] [*override*] [*flag*]

.TC—table of contents

.TC [*slevel*] [*spacing*] [*tlevel*] [*tab*] [*head1*] [*head2*] [*head3*] [*head4*] [*head5*]

.TE—table end

.TH—table header

.TH [*N*]

.TL—title of memorandum

.TL [*charging-case*] [*filing-case*]

.TM—Technical Memorandum number(s)

.TM [*number*]

*****.TP**—top-of-page macro

.TS—table start

.TS [*H*]

*****.TX**—table of contents user exit

*****.TY**—table of contents user exit

.VL—variable-item list start

.VL *text-indent* [*mark-indent*]

.VM—vertical margins

.VM [*top*] [*bottom*]

.WC—footnote and display width control

.WC [*format*]

Corporate Headquarters
Zilog, Inc.
1315 Dell Avenue
Campbell, CA 95008 USA
Phone: 408 370 8000
TWX: 910 338 7621

United Kingdom
Zilog (U.K.) Limited
Zilog House
43-53 Moorbridge Road
Maidenhead
Berkshire, SL6 8PL England
Phone: 0628 392000
Telex: 848609

France
Zilog S.A.R.L.
31, Place des Corolles
Cedex 31
92098 Paris La Defense
France
Phone: 1 334 60 09
Telex: 611445F

West Germany
Zilog GmbH
Eschenstrasse 8
D-8028 TAUFKIRCHEN
Munich, West Germany
Phone: 89 612 6046
Telex: 529110 Zilog d

03-3269 Printed in USA